



Introduction to Online Control

version 1.1 β

Elad Hazan

Karan Singh



We are in a golden age for control and decision making. A proliferation of new applications including self-driving vehicles, humanoid robots, and artificially intelligent drones open a new set of challenges for control theory to address. Hazan and Singh have written the definitive book on the New Control Theory—non-stochastic control. The phrase “a paradigm shift” has become cliché from overuse, but here it is truly well deserved; the authors have revisited the foundations by focusing on building controllers that perform nearly as well as if they knew future disturbances in advance, rather than relying on probabilistic or worst-case models. The non-stochastic control approach has extended one of the most profound ideas in mathematics of the 20th century, online (no-regret) learning, to master sequential decision making with continuous actions. This leads to high performance in benign environments and resilience in adversarial ones. The book, authored by pioneers in the field, presents both foundational concepts and the latest research, making it an invaluable resource.

— Drew Bagnell

As someone who has worked extensively on learning theory and online learning, and later applied these ideas in domains such as autonomous driving and humanoid robotics, I find this book both timely and inspiring. It introduces a regret-minimization framework for control that draws on the elegance and power of online learning. Traditional control theory often models noise either as stochastic—sometimes unrealistically optimistic—or adversarial—often overly conservative. This book charts a new path by asking a deeper question: while we cannot predict noise, can we perform nearly as well as if we could? The answer, developed here, is a novel and exciting paradigm that bridges learning theory and control, and I believe it will have a lasting impact on both research and practice.

— Shai Shalev-Shwartz

To my family: Dana, Hadar, Yoav, Oded, and Deluca,
—EH

To my parents— A & V.
—KS

Preface

This text presents an introduction to an emerging paradigm in control of dynamical systems and differentiable reinforcement learning called *online nonstochastic control*. The new approach applies techniques from online convex optimization and convex relaxations to obtain new methods with provable guarantees for classical settings in optimal and robust control.

The primary distinction between online nonstochastic control and other frameworks is the objective. In optimal control, robust control, and other control methodologies that assume stochastic noise, the goal is to perform comparably to an offline optimal strategy. In online nonstochastic control, both the cost functions and the perturbations from the assumed dynamical model are chosen by an adversary. Thus, the optimal policy is not defined a priori. Rather, the goal is to attain low regret against the best policy in hindsight from a benchmark class of policies.

This objective suggests the use of the decision-making framework of online convex optimization as an algorithmic methodology. The resulting methods are based on iterative mathematical optimization algorithms and are accompanied by finite-time regret and computational complexity guarantees.

Acknowledgments

This book was a journey that started in Princeton at the Google DeepMind Lab in the summer of 2018. At that time Sham Kakade visited the group, which consisted of the authors together with Naman Agarwal, Brian Bullins, Xinyi Chen, Cyril Zhang, and Yi Zhang, henceforth *the gang*. The *gang* worked on various aspects of optimization, reinforcement learning, and dynamical systems. The first paper, in which online nonstochastic control was basically invented, started in room 401 in the computer science building at Princeton University. We remember and cherish the exact moment along with the mathematical writings in white chalk on that blackboard.

Ever since then, the gem of an idea turned out into a complete theory that is now mainstream in machine learning and control, with numerous applications and refinements. Notable events that contributed to this theory include the following. Max Simchowitz, who was a graduate student then, visited Princeton during 2019-2020, where he collaborated with the group, and the theory was extended to partial observed systems. Paula Gradu, who was then an undergraduate, and Edgar Minasyan, while a graduate student, contributed to the study of time-varying and nonlinear dynamical systems. Anirudha Majumdar joined for several important projects, notably for iterative linear control. Udaya Ghai and Arushi Gupta studied extending the GPC to model-free RL.

Most recently, this theory has been used to design new architectures for sequence prediction. The theory of spectral filtering was extended by the aforementioned members of the group, and newer folks joined, including Daniel Suo and Annie Marsden.

This book was further refined through teaching. Elad Hazan gratefully acknowledges the numerous contributions and insights of the students of the course *computational control theory* delivered at Princeton during COVID over Zoom during 2020-2021. The text was further enhanced by students of *theoretical machine learning* taught at Princeton University in the fall of 2024-2025.

The following colleagues and students were particularly helpful, making insightful comments, finding various typos and suggesting improvements: Gon Buzaglo, Anand Brahmbhatt, Sofia Druchyna, Jonathan Pillow, and many more.

Throughout the writing of this book, we engaged in extensive discussions with ChatGPT, an AI assistant. Although it does not claim authorship, its ability to refine mathematical reasoning and clarify complex topics has been a remarkable collaboration in human-AI interaction. In fact, it wrote this acknowledgment all by itself!

List of Symbols

General

$\stackrel{\text{def}}{=}$	definition
$\arg \min\{\}$	the argument minimizing the expression in braces
$[n]$	the set of integers $\{1, 2, \dots, n\}$
$\mathbf{1}_A$	the indicator function, equals one if A is true, else zero

Linear Algebra, Geometry and Calculus

$\mathbb{R}^d$	d dimensional Euclidean space
Δ_d	d dimensional simplex, $\{\sum_i \mathbf{x}_i = 1, \mathbf{x}_i \geq 0\}$
$\mathbb{S}$	d dimensional sphere, $\{\ \mathbf{x}\ = 1\}$
$\mathbb{B}$	d dimensional ball, $\{\ \mathbf{x}\ \leq 1\}$
$\mathbb{R}$	real numbers
$\mathbb{C}$	complex numbers
$ A $	determinant of matrix A
$\text{Tr}(A)$	trace of matrix A
$A \succeq 0$	A is positive semi-definite, $\forall v, v^\top A v \geq 0$
$A \succeq B$	$A - B$ is positive semi-definite

Control

f	dynamics function
$\mathbf{x}_t \in \mathbb{R}^{d_{\mathbf{x}}}$	state at time t
$\mathbf{u}_t \in \mathbb{R}^{d_{\mathbf{u}}}$	control at time t
$\mathbf{w}_t \in \mathbb{R}^{d_{\mathbf{w}}}$	perturbation at time t
$\mathbf{y}_t \in \mathbb{R}^{d_{\mathbf{y}}}$	observation at time t
A_t, B_t, C_t	system matrices for linear dynamical system
γ	BIBO stabilizability $\ \mathbf{w}_t\ \leq 1 \longrightarrow \ \mathbf{x}_t\ \leq \gamma$
h	Number of parameters in a policy class
κ	norm bound on policy parameters
K_t	stabilizing linear controller at time t
$K_r(A, B)$	Kalman controllability matrix of order r
$\hat{K}(A, C)$	Kalman observability matrix

Reinforcement learning

S, A, γ, R, P	Markov Decision Process with states S , actions A , discount factor γ , rewards R and transition probability matrix P
R_t	reward at time t
$\mathbf{P}_{ss'}^a$	probability of transition from s to s' given action a
$\mathbf{v} : S \rightarrow \mathbb{R}$	value function
$Q : S \times A \rightarrow \mathbb{R}$	Q function
$\pi : S \rightarrow A$	policy

Optimization

$\mathbf{x}$	vectors in the decision set
$\mathcal{K}$	decision set
$\nabla^k f$	the k 'th differential of f ; note $\nabla^k f \in \mathbb{R}^{d^k}$
$\nabla^{-2} f$	the inverse Hessian of f
∇f	the gradient of f
∇_t	the gradient of f at point $\mathbf{x}_t$
$\mathbf{x}^*$	the global or local optima of objective f
h_t	objective value distance to optimality, $h_t = f(\mathbf{x}_t) - f(\mathbf{x}^*)$
d_t	Euclidean distance to optimality $d_t = \ \mathbf{x}_t - \mathbf{x}^*\ $
G	upper bound on norm of subgradients
D	upper bound on Euclidean diameter
D_p, G_p	upper bound on the p -norm of the subgradients/diameter

Contents

Preface	v
Acknowledgments	vii
List of Symbols	ix
 I Background in Control and Reinforcement Learning	 1
1 Introduction	3
1.1 What is This Book About?	3
1.2 The Origins of Control	4
1.3 Formalization and Examples of a Control Problem	5
1.3.1 Example: Control of a Medical Ventilator	6
1.4 Simple Control Algorithms	7
1.4.1 The Bang-Bang Controller	7
1.4.2 The PID Controller	8
1.5 Classical Theory: Optimal and Robust Control	8
1.6 The Need for a New Theory	9
1.6.1 Online Nonstochastic Control Theory	10
1.6.2 A New Family of Algorithms	11
1.7 Bibliographic Remarks	13
 2 Dynamical Systems	 15
2.1 Examples of Dynamical Systems	15
2.1.1 Medical Ventilator	15
2.1.2 Double Integrator	16
2.1.3 Pendulum Swing-Up	17
2.1.4 Autoregressive Language Models	18
2.1.5 Epidemiological Models	19

2.2	Solution Concepts for Dynamical Systems	20
2.2.1	Equilibrium of Dynamical Systems	20
2.2.2	Stabilizability and Controllability	21
2.2.3	Quantitative Definitions of Stabilizability	22
2.3	Complexity of Dynamical Systems and Hardness of Stabilizability	22
2.3.1	Computational Hardness of Stabilizability	23
2.4	Bibliographic Remarks	26
2.5	Exercises	27
3	Markov Decision Processes	29
3.1	Reinforcement Learning	29
3.1.1	A Classical Example: Bubbo's World	31
3.2	Markov Decision Processes	33
3.2.1	Markov Chains	33
3.2.2	Markov Reward Processes	35
3.2.3	Markov Decision Processes	37
3.3	The Bellman Equation	39
3.3.1	Linear Programming Formulation	40
3.4	The Value Iteration Algorithm	41
3.5	Bibliographic Remarks	44
3.6	Exercises	45
4	Linear Dynamical Systems	47
4.1	Approximating General Dynamics via LTV Systems	48
4.2	Stabilizability of Linear Systems	49
4.2.1	Efficient Verification of Stabilizability	50
4.3	Controllability of Linear Dynamical Systems	51
4.4	Quantitative Definitions	53
4.4.1	Stabilizability of Time-Invariant Systems	53
4.4.2	Stabilizability of Time-Varying Systems	54
4.4.3	Controllability of Time-Invariant Systems	54
4.5	Bibliographic Remarks	56
4.6	Exercises	57
5	Optimal Control of Linear Dynamical Systems	59
5.1	The Linear Quadratic Regulator	59
5.2	Optimal Solution of the LQR	60
5.3	Infinite Horizon LQR	62
5.3.1	Solving the DARE	64

5.4	Robust Control	64
5.5	Bibliographic Remarks	66
5.6	Exercises	67

II Basics of Online Control 69

6 Regret in Control 71

6.1	Online Convex Optimization	72
6.1.1	Online Gradient Descent	72
6.2	Regret for Control	74
6.3	Expressivity of Control Policy Classes	76
6.3.1	Linear Policies	76
6.3.2	Disturbance Action Controllers	77
6.4	Bibliographic Remarks	82
6.5	Exercises	83

7 Online Nonstochastic Control 85

7.1	From Optimal and Robust to Online Control	86
7.1.1	A Shift to Online Nonstochastic Control	87
7.2	The Online Nonstochastic Control Problem	88
7.2.1	Improper Learning and Convex Relaxation	89
7.3	The Gradient Perturbation Controller	90
7.3.1	Convexity of the Losses	92
7.3.2	Loss Functions with Memory	93
7.4	Bibliographic Remarks	95
7.5	Exercises	96

8 Online Nonstochastic System Identification 97

8.1	The Role of System Identification in Online Control	97
8.2	A Nonstochastic Approach to System Identification	98
8.3	Nonstochastic System Identification	99
8.4	From Identification to Nonstochastic Control	102
8.5	Summary and Key Takeaways	103
8.5.1	Looking Ahead	103
8.6	Bibliographic Remarks	105
8.7	Exercises	106

III	Learning and Filtering	107
9	Learning in Unknown Linear Dynamical Systems	109
9.1	Learning in Dynamical Systems	110
9.2	Online Learning of Dynamical Systems	111
9.3	Classes of Prediction Rules	111
9.3.1	Linear Predictors	112
9.3.2	Linear Dynamical Predictors	112
9.4	Efficient Learning of Linear Predictors	114
9.5	Summary	115
9.6	Bibliographic Remarks	117
9.7	Exercises	118
10	Kalman Filtering	119
10.1	Observable Systems	120
10.2	Optimality of the Kalman Filter	121
10.2.1	The Power of Linear Predictors	121
10.2.2	Closed-Loop Optimal Predictors	122
10.2.3	Tools from Linear Regression	123
10.2.4	Analysis	123
10.2.5	Infinite Horizon Kalman Filtering	125
10.2.6	Linear Predictors and the Kalman Filter	125
10.3	Bayes-Optimality under Gaussian Noise	125
10.4	Conclusion	127
10.5	Bibliographic Remarks	128
10.6	Exercises	129
11	Spectral Filtering	131
11.1	Spectral filtering in One Dimension	132
11.1.1	The Magic of Hankel Matrices	133
11.2	Spectral Predictors	134
11.2.1	Generalization to High Dimension	138
11.3	Online Spectral Filtering	138
11.4	Conclusion	140
11.5	Bibliographic Remarks	142
11.6	Exercises	144
12	Luenberger Observers	145
12.1	Lower Bounds for Learning Asymmetric LDS	145
12.2	Pole Placement	147

12.3	The Luenberger Observer	149
12.3.1	Luenberger Complexity	149
12.3.2	One Dimensional Example	151
12.4	Reduction to Stable Learning	151
12.4.1	A Learnable Spectral Region	151
12.4.2	The Luenberger Program	152
12.4.3	Regret Reduction	152
12.5	Summary	154
12.6	Bibliographic Remarks	156
12.7	Exercises	157
13	Learning in Nonlinear Dynamical Systems	159
13.1	Global Linear Approximation via Discretization	160
13.1.1	The ε -Net Construction	160
13.1.2	Approximation Guarantee	161
13.2	The Koopman Operator	163
13.3	Improper Learning of Nonlinear Dynamics	163
13.4	Bibliographic Remarks	166
13.5	Exercises	168
IV	Online Control with Partial Observation	169
14	Policy Classes for Partially Observed Systems	171
14.1	Linear Observation Action Controllers	172
14.1.1	Generalized Linear Observation Action Controllers	173
14.2	Linear Dynamic Controllers	174
14.3	Disturbance Response Controllers	175
14.3.1	Expressivity of Disturbance Response Controllers	176
14.4	Summary	179
14.5	Bibliographic Remarks	180
14.6	Exercises	181
15	Online Nonstochastic Control with Partial Observation	183
15.1	The Gradient Response Controller	184
15.1.1	Finishing up the Regret Bound	187
15.2	Extension to Time Varying Systems	187
15.3	Conclusion	188
15.4	Bibliographic Remarks	190
15.5	Exercises	191

A	Second Order Online Learning with the Square Loss	193
A.1	Setting	193
A.2	Algorithm: ONS for Square Loss	194
A.3	Regret Bound	195
A.4	The VAW Algorithm for Online Least Squares	197
A.4.1	The VAW Forecaster	197
A.4.2	VAW with Known Additive Offsets	199
	Bibliography	203

Part I

Background in Control and Reinforcement Learning

Chapter 1

Introduction

1.1 What is This Book About?

Control theory is the engineering science of manipulating physical systems to achieve the desired functionality. Its history is centuries old and spans rich areas from the mathematical sciences. A brief list includes differential equations, operator theory and linear algebra, functional analysis, harmonic analysis, and more. However, it mostly does not concern modern algorithmic theory or computational complexity from computer science.

The advance of the deep learning revolution has brought to light the power and robustness of algorithms based on gradient methods. The following quote is attributed to deep learning pioneer Yann Lecun during a Barbados workshop on deep learning theory circa February 2020.

“Control = reinforcement learning with gradients.”

This book concerns this exact observation: *How can we exploit the differentiable structure in natural physical environments to create efficient and robust gradient-based algorithms for control?*

The answer we give is **not** to apply gradient descent to any existing algorithm. Rather, we rethink the paradigm of control from the start. What can be achieved in a given dynamical system, whether known or unknown, fully observed or not, linear or nonlinear, and without assumptions on the noise sequence?

This is a deep and difficult question that we fall short of answering. However, we do propose a different way of answering this question: we define the control problem from the point of view of an online player in a repeated game. This gives rise to a simple but powerful framework called nonstochastic control theory. The framework is a natural fit to applying techniques

from online convex optimization, resulting in new, efficient, gradient-based control methods.

1.2 The Origins of Control

The industrial revolution was in many ways shaped by the steam engine that came to prominence in the eighteenth century. One of its most critical components was introduced by the Scottish inventor James Watt: the *centrifugal governor*, which is amongst the earliest examples of a control mechanism. A major source of efficiency losses of the earlier designs was the difficulty in regulating the engine. Too much or too little steam could easily cause engine failure. The centrifugal governor was created to solve this problem.

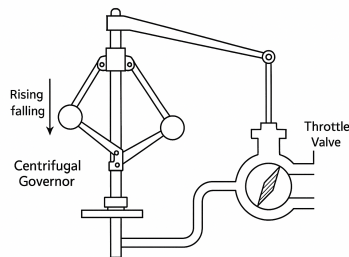


Figure 1.1: A centrifugal governor.

The centrifugal governor consists of a shaft connected to two metal balls that are allowed to rotate. The increased shaft speed causes the balls to spin faster and the resulting centrifugal force causes the balls to lift. The balls are in turn attached to a pin that controls the amount of steam entering the engine, so as the balls spin faster, the pin lifts and limits the amount of fuel that can enter the system. However, if the balls spin slower, the pin is lowered, allowing more steam to enter the system. This provides a natural, direct, and proportional control to the system.

The impact of the centrifugal governor on the steam engine, and in turn on the industrial revolution, illustrates the importance of the field of control in engineering and the sciences. Since the eighteenth century, our world has changed quite a bit, and the importance of control has only grown.

The challenges of controlling systems have moved from the steam engine to the design of aircraft, robotics, and autonomous vehicles. The importance of designing efficient and robust methods remains as crucial as in the industrial revolution.

1.3 Formalization and Examples of a Control Problem

With the physical example of the centrifugal governor in mind, we give a mathematical definition of a control problem. The first step is to formally define a dynamical system in the following way.

Definition 1.1 (Dynamical system). *A dynamical system evolving from an initial state $\mathbf{x}_1$ is given by the following equations:*

$$\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t), \quad \mathbf{y}_t = g_t(\mathbf{x}_t).$$

Here $\mathbf{x}_t \in \mathbb{R}^{d_x}$ is the state of the system, $\mathbf{y}_t \in \mathbb{R}^{d_y}$ is the observation, $\mathbf{u}_t \in \mathbb{R}^{d_u}$ is the control input, and $\mathbf{w}_t \in \mathbb{R}^{d_w}$ is the perturbation.

The function g_t is a mapping from the state to the observed state, and the function f_t is the transition function, given the current state $\mathbf{x}_t$, the input control $\mathbf{u}_t$, and the perturbation $\mathbf{w}_t$. The subscript indicates the relevant quantities at time step t .

Given a dynamical system, there are several possible objectives that we can formulate. We give a generic formulation below that captures many common control formulations, notably the optimal control problem and the robust control problem. These will be addressed in much more detail later in this book. In this problem formulation, the goal is to minimize the long-term horizon cost given by a sequence of cost functions.

Definition 1.2 (A generic control problem). *Given a dynamical system with starting state $(\mathbf{x}_1, \mathbf{y}_1)$, iteratively construct controls $\mathbf{u}_t$ to minimize a sequence of loss functions $c_t(\mathbf{y}_t, \mathbf{u}_t)$, $c_t: \mathbb{R}^{d_y+d_u} \rightarrow \mathbb{R}$. This is given by the following mathematical program:*

$$\begin{aligned} \min_{\mathbf{u}_{1:T}} \quad & \sum_t c_t(\mathbf{y}_t, \mathbf{u}_t) \\ \text{s.t.} \quad & \mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t), \quad \mathbf{y}_t = g_t(\mathbf{x}_t). \end{aligned}$$

At this point, many aspects of the generic problem have not been specified. These include:

- Are the transition functions and observation functions known or unknown to the controller?
- How accurately is the transition function known?

- What is the mechanism that generates the perturbations $\mathbf{w}_t$?
- Are the cost functions known or unknown ahead of time?

Different theories of control differ by the answers they give to the above questions, and these in turn give rise to different methods.

Instead of postulating a hypothesis about how our world should behave, we proceed to give an intuitive example of a control problem. We can then reason about the different questions and motivate the nonstochastic control setting.

1.3.1 Example: Control of a Medical Ventilator

A recent relevant example of control is in the case of medical ventilators. The ventilator connects the trachea in the lungs to a central pressure chamber, which helps maintain positive-end expiratory pressure (PEEP) to prevent the lungs from collapsing and simulate healthy breathing in patients with respiratory problems.

The ventilator must take into account the structure of the lung to determine the optimal pressure to induce. Such structural factors include *compliance*, or the change in lung volume per unit pressure, and *resistance*, or the change in pressure per unit flow.

A simplistic formalization of the dynamical system of the ventilator and lung can be derived from the physics of a connected two-balloon system. The dynamics equations can be written as

$$\begin{aligned} p_t &= C_0 + C_1 v_t^{-1/3} + C_2 v_t^{5/3}, \\ v_{t+1} &= v_t + \Delta_t u_t, \end{aligned}$$

where p_t is the observed pressure, v_t is the volume of the lung, u_t is the input flow (control) and Δ_t is the unit of time resolution of the device. C_0 , C_1 , and C_2 are constants that depend on the particular system/lung-ventilator pair.

The goal of ventilator control is to regulate the pressure according to some predetermined pattern. As a dynamical system, we can write $v_{t+1} = f(v_t, u_t, w_t)$, $p_t = g(v_t)$, and we can define the cost function to be $c_t(p_t, u_t) = |p_t - p_t^*|^2$, where p_t^* is the desired pressure at time t . The goal, then, is to choose a sequence of control signals $\{u_t\}_{t=1}^T$ to minimize the total cost over T iterations.

In figure 1.2 we see an illustration of a simple controller, which we will soon describe, called the PID controller, as it performs on a ventilator and lung simulator.

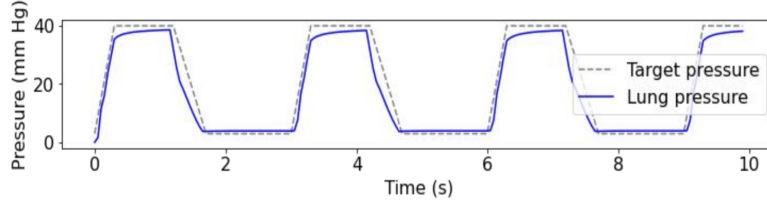


Figure 1.2: Performance of the PID controller on a mechanical ventilator, from [SZG⁺21].

1.4 Simple Control Algorithms

The previous section detailed the importance of control problems, specifically through the example of a critical care medical device. In this section, we detail methods for controlling a medical ventilator, or indeed any physical device that can be modeled as a dynamical system.

The following methods are simplistic, but still widely used. For example, the PID algorithm is still used in most medical ventilators today. In later sections, we describe a modern optimization-based framework that yields more sophisticated algorithms that are capable of better performance.

1.4.1 The Bang-Bang Controller

Bang-bang control is one of the simplest control strategies. Consider a scalar dynamical system, where it is desirable to keep the state x_t within a prespecified dynamic range $[x_{\min}, x_{\max}]$ (which can vary over time) and where the scalar control input u_t must at all times be within the interval $[u_{\min}, u_{\max}]$, which is assumed to contain zero for simplicity. A bang-bang controller chooses the control input at time t as

$$u_t = \mathbf{1}_{\{x_t < x_{\min}\}} \times u_{\max} + \mathbf{1}_{\{x_t > x_{\max}\}} \times u_{\min}.$$

A bang-bang controller saturates its inputs to extreme values in an attempt to confine the state of the system to a desirable range. For example, in the context of mechanical ventilation, one could choose $[p_t^* - \Delta p, p_t^* + \Delta p]$ as a reasonable region around the prescribed pressure p_t^* as a desirable range for p_t . In such a case, a bang-bang controller would allow oxygen to enter at the highest possible rate whenever the pressure drops below $p_t^* - \Delta p$.

This discussion highlights two unsatisfactory aspects of the bang-bang control scheme: one, it ignores the cost associated with acting with the

extreme values of the control input; two, it produces oscillatory and rapidly changing behavior in the control state.

1.4.2 The PID Controller

A more sophisticated control algorithm is the PID control method. Consider a control method that chooses the control to be a linear function of the observed states, that is:

$$\mathbf{u}_t = \sum_{i=0}^k A_i \mathbf{x}_{t-i},$$

for some coefficients A_i , where we consider only the $k+1$ most recent states. One of the most useful special cases of this linear family is the PID controller.

A special subfamily of linear controllers is that of linear controllers that optimize on a predefined basis of historical observations. More precisely,

1. Proportional Control (e.g. Centrifugal Governor): $\mathbf{u}_t = \alpha_0 \mathbf{x}_t$.
2. Integral Control: $\mathbf{u}_t = \beta \sum_{i < t} \mathbf{x}_{t-i}$.
3. Derivative Control: $\mathbf{u}_t = \gamma(\mathbf{x}_t - \mathbf{x}_{t-1})$.

The PID controller, named after the nature of the three components that make up it, specifies three coefficients α, β, γ to generate a control signal. In that sense, it is extremely sparse, which from a learning-theoretic perspective is a good indication of generalization.

In fact, the PID controller is the method of choice for numerous engineering applications, including control of medical ventilators.

1.5 Classical Theory: Optimal and Robust Control

Classical theories of control differ within the generic control problem formulation 1.2 primarily in one significant aspect: the perturbation model. A rough divide between the two main theories – *optimal control* and *robust control* – is as follows.

Optimal control postulates a probabilistic model for the perturbations, which are typically i.i.d. draws from a Gaussian distribution. However, other types of distribution can also be considered. As such, the control problem

can be rephrased in a more precise stochastic mathematical program,

$$\begin{aligned} \min_{\mathbf{u}_{1:T} \mathbf{w}_{1:T}} \mathbf{E} \left[\sum_t c_t(\mathbf{y}_t, \mathbf{u}_t) \right] \\ \text{s.t. } \mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t), \quad \mathbf{y}_t = g(\mathbf{x}_t). \end{aligned}$$

In contrast, robust control theory plans for the worst-case noise from a given set of constraints, denoted $\mathcal{K}$. Thus, the robust control problem can be rephrased as the following mathematical program,

$$\begin{aligned} \min_{\mathbf{u}_{1:T}} \max_{\mathbf{w}_{1:T} \in \mathcal{K}} \left[\sum_t c_t(\mathbf{y}_t, \mathbf{u}_t) \right] \\ \text{s.t. } \mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t), \quad \mathbf{y}_t = g(\mathbf{x}_t). \end{aligned}$$

The two formulations have been debated for decades, with pros and cons for each side. Optimal control focuses on the average case and often yields simpler formulations for optimization, resulting in more efficient methods. Robust control, in contrast, allows for more difficult noise models and some amount of model misspecification. However, it is too pessimistic for many natural problems.

Nonstochastic control theory takes the best of both worlds, simultaneously granting robustness to adversarial noise and allowing the use of optimistic behavior when appropriate while providing computationally efficient methods.

1.6 The Need for a New Theory

Consider the problem of flying a drone from the source to the destination subject to unknown weather conditions. The aerodynamics of flight can be modeled sufficiently well by time-varying linear dynamical systems, and existing techniques are perfectly capable of doing a great job for indoor flight. However, the wind conditions, rain, and other uncertainties are a different story. Certainly, the wind is not an i.i.d. Gaussian random variable! Optimal control theory, which assumes this zero-mean noise, is therefore overly optimistic and inaccurate for our problem.

The designer might resort to robust control theory to account for all possible wind conditions, but this is overly pessimistic. What if we encounter a bright sunny day after all? Planning for the worst case would mean slow and fuel-inefficient flight.

We would like an adaptive control theory that allows us to attain the best of both worlds: an efficient and fast flight when the weather permits, and a careful drone maneuvering when this is required by the conditions. Can we design a control theory that will allow us to take into account the specific instance perturbations and misspecifications, and give us finite-time provable guarantees? This is the subject of online nonstochastic control!

1.6.1 Online Nonstochastic Control Theory

This book is concerned with nonstochastic noise. When dealing with nonstochastic, that is, arbitrary or even adversarial perturbations, the optimal policy, that is, a decision making rule mapping observations to control inputs, is not clear a priori. Rather, an optimal policy for the observed perturbations can be determined *only in hindsight*. This is a significant difference from optimal and robust control theory, where the mapping from states to actions can be completely predetermined based upon the noise model.

Since optimality is defined only in hindsight, the goal is not predetermined. We thus shift to a different performance metric borrowed from the world of learning in repeated games, namely regret. Specifically, we consider regret with respect to a reference class of policies. In online control, the controller iteratively chooses a control $\mathbf{u}_t$. The controller then observes the next state of the system $\mathbf{x}_{t+1}$ and suffers a loss of $c_t(\mathbf{x}_t, \mathbf{u}_t)$ according to an adversarially chosen loss function. For simplicity, we consider the case of full state observation where $\mathbf{y}_t = \mathbf{x}_t$. Let $\Pi = \{\pi: \mathbf{x} \rightarrow \mathbf{u}\}$ be a class of policies. The regret of the controller with respect to Π is defined as follows.

Definition 1.3. *Given a dynamical system, the regret of an online control algorithm $\mathcal{A}$ over T iterations with respect to a class of policies Π is given by:*

$$\text{regret}_T(\mathcal{A}, \Pi) = \max_{\mathbf{w}_{1:T}: \|\mathbf{w}_t\| \leq 1} \left(\sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi \in \Pi} \sum_{t=1}^T c_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi) \right),$$

where $\mathbf{u}_t = \mathcal{A}(\mathbf{x}_t)$ are the controls generated by $\mathcal{A}$, and $\mathbf{x}_t^\pi, \mathbf{u}_t^\pi$ are the counterfactual state sequence and controls under the policy π , that is,

$$\begin{aligned} \mathbf{u}_t^\pi &= \pi(\mathbf{x}_t^\pi) \\ \mathbf{x}_{t+1}^\pi &= f_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi, \mathbf{w}_t). \end{aligned}$$

Henceforth, if T, Π and $\mathcal{A}$ are clear from the context, we drop these symbols when defining regret.

At this point, the reader may wonder why we should compare to a reference class, as opposed to all possible policies. There are two main answers to this question:

1. First, the best policy in hindsight out of all possible policies may be very complicated to describe and to reason about. This argument was made by the mathematician Tyrell Rockafellar; see the bibliographic section for more details.
2. Secondly, it can be shown that it is impossible, in general, to obtain sublinear regret versus the best policy in hindsight. Rather, a different performance metric called *competitive ratio* can be analyzed, as we discuss later in the book.

The non-stochastic control problem can now be stated as finding an efficient algorithm that minimizes the worst-case regret vs. meaningful policy classes that we examine henceforth. It is important to mention that *the algorithm does not have to belong to the comparator set of policies!* Indeed, in the most powerful results, we will see that the algorithm will learn and operate over a policy class that is strictly larger than the comparator class.

1.6.2 A New Family of Algorithms

The policy regret minimization viewpoint in control leads to a new approach to algorithm design. Instead of a priori computation of the optimal controls, online convex optimization suggests modifying the controller according to the costs and dynamics, in the flavor of adaptive control, in order to achieve provable bounds on the policy regret.

As a representative example, consider the *Gradient Perturbation Controller* described in Algorithm 1. The algorithm maintains matrices that are used to create the control as a linear function of past perturbations, according to line 3. These parameters change over time according to a gradient-based update rule, where the gradient is calculated as the derivative of a counterfactual cost function with respect to the parametrization, according to line 5.

The version given here assumes that the time-invariant dynamics f are known. The GPC computes a control that is a linear function of past perturbations. It modifies the linear parameters according to the loss function via an iterative gradient method. In Chapter 7, we prove that this algorithm attains sublinear regret bounds for a large class of time-varying and time-invariant dynamical systems. In later chapters, we explore extensions

Algorithm 1 Gradient Perturbation Controller(GPC) - simplified version

- 1: Input: h, η , initial parameters $M_{1:h}^1$, dynamics f .
 - 2: **for** $t = 1 \dots T$ **do**
 - 3: Use control $\mathbf{u}_t = \sum_{i=1}^h M_i^t \mathbf{w}_{t-i}$
 - 4: Observe state $\mathbf{x}_{t+1}$, compute perturbation $\mathbf{w}_t = \mathbf{x}_{t+1} - f(\mathbf{x}_t, \mathbf{u}_t)$
 - 5: Construct loss $\ell_t(M_{1:h}) = c_t(\mathbf{x}_t(M_{1:h}), \mathbf{u}_t(M_{1:h}))$
 Update $M_{1:h}^{t+1} \leftarrow M_{1:h}^t - \eta \nabla \ell_t(M_{1:h}^t)$
 - 6: **end for**
-

to nonlinear dynamical systems and unknown systems, which lie at the heart of nonstochastic control theory.

1.7 Bibliographic Remarks

Control theory is a mature and active discipline with both deep mathematical foundations and a rich history of applications. In terms of application, self-regulating feedback mechanisms that limit the flow of water, much like the modern float valve, were reportedly used in ancient Greece over two thousand years ago. The first mathematical analysis of such systems appeared in a paper by James Clerk Maxwell on the topic of governors [Max68], shortly after Maxwell had published his treatise on the unification of electricity and magnetism. For an enlightening historical account, see [FCZI03].

For a historical account on the development of the PID controller, see [Ben93]. The theoretical and practical properties of this controller are described in detail in [ÅH95].

There are many excellent textbooks on control theory, which we point to in later chapters. The book of [Ber07] offers extensive coverage of topics from the point of view of dynamic programming. The texts [Ste94, ZDG96a], in contrast, delve deeply into results on linear control, the most well-understood branch of control theory. Rockafellar [Roc87] proposed the use of convex cost functions to model the cost in a linear dynamical system to incorporate constraints on state and control and suggested computational difficulties with this approach. The text [SL91] gives a thorough introduction to adaptive control theory.

Online nonstochastic control theory and the Gradient Perturbation Control algorithm were proposed in [ABH⁺19]. A flurry of novel methods and regret bounds for online nonstochastic control have appeared in recent years, many of which are surveyed in [HS21]. Notable among them are logarithmic regret algorithms [AHS19, FS20, Sim20], nonstochastic control for unknown systems [HKS20], black-box online nonstochastic control [CH21], and nonstochastic control with partial observability [SSH20].

The theory is heavily based on recent developments in decision making and online learning, notably the framework of online convex optimization [Haz16], and especially its extensions to learning with memory [AHM13].

Chapter 2

Dynamical Systems

In this chapter, we return to the general problem of control from Definition 1.2: choosing a sequence of controls to minimize a sequence of cost functions. Before moving on to algorithms for control, it is useful to ask what kinds of goals are natural and what kinds of guarantees are even possible for a dynamical system.

To that end, we consider several examples of dynamical systems and introduce a few basic solution concepts, including equilibrium, stability, stabilizability, and controllability. These examples also illustrate the kinds of observation models and control objectives that arise in real applications.

2.1 Examples of Dynamical Systems

2.1.1 Medical Ventilator

In the task of mechanical ventilation, the control algorithm dictates the amount of oxygen inflow u_t (in m^3/s) into the lungs of a patient. This inflow approximately determines the volume of the lungs v_t via the simple relation

$$v_{t+1} = v_t + \Delta u_t,$$

where Δ is the time elapsed between successive actions of the discrete-time control algorithm. Being difficult to directly measure or estimate, the quantity v_t is a hidden or latent variable from the controller's perspective. Instead, the controller observes the air pressure p_t . A simplified physical model relating pressure and volume is

$$p_t = C_0 + C_1 v_t^{-1/3} + C_2 v_t^{5/3}.$$

This relation can be derived from stress–strain considerations by modeling the left and right lungs, connected by the trachea, as in a two-balloon experiment.

The objective of the control algorithm is to follow an ideal pressure trajectory p_t^* prescribed by the physician as closely as possible, for example via the stage cost

$$c_t(p_t, u_t) = |p_t - p_t^*|^2 + |u_t|^2,$$

while penalizing large airflow rates so as to limit patient discomfort. See Figure 2.1 for a schematic representation of airflow when a lung is connected to a mechanical ventilator.

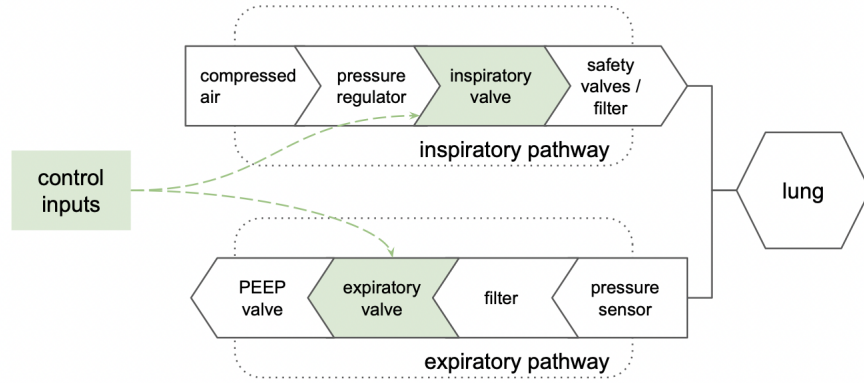


Figure 2.1: A schematic of the respiratory circuit from [SZG⁺21].

2.1.2 Double Integrator

The double integrator is a common example of a *linear* dynamical system. In its simplest form, it models a one-dimensional point object moving along the real line. The state $\mathbf{x}_t \in \mathbb{R}^2$ is two-dimensional, with the first coordinate indicating position and the second denoting velocity. The control algorithm chooses a scalar actuation $u_t \in \mathbb{R}$ that directly affects the velocity coordinate at each time step; after an appropriate normalization, u_t may be interpreted as a force or acceleration input applied over one sampling interval. The dynamics equation can be written as

$$\mathbf{x}_{t+1} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \mathbf{x}_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t + \mathbf{w}_t.$$

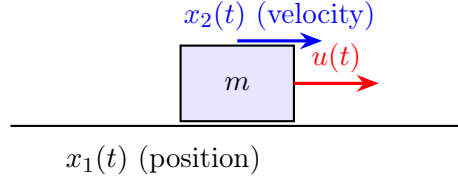


Figure 2.2: Double integrator illustration showing state coordinate $x_1(t)$ (position), state coordinate $x_2(t)$ (velocity), and control input $u(t)$.

The position is linearly related to the velocity through the sampling period Δ . Optionally, the state is also affected by external perturbations $\mathbf{w}_t$ outside the influence of the controller.

The goal here is to ensure that the point object arrives and then comes to rest at some coordinate $x^* \in \mathbb{R}$ as quickly as possible, while also not expending too much energy. A natural stage cost is

$$c(\mathbf{x}_t, u_t) = \left\| \mathbf{x}_t - \begin{bmatrix} x^* \\ 0 \end{bmatrix} \right\|^2 + u_t^2.$$

2.1.3 Pendulum Swing-Up

The swing-up pendulum is one of the simplest examples of rotational motion. It consists of a point mass m attached to the end of a massless rod of length l , with the other end fixed at a pivot. The system is subject to gravity with acceleration g , and the controller applies a torque $u_t \in \mathbb{R}$ at each time step. If θ_t denotes the angle of the rod measured from the downward vertical, then a discretization of the continuous-time dynamics gives

$$\theta_{t+1} = \theta_t + \Delta \dot{\theta}_t, \quad \dot{\theta}_{t+1} = \dot{\theta}_t + \Delta \left(\frac{u_t}{ml^2} - \frac{g}{l} \sin \theta_t \right).$$

Thus the state is $\mathbf{x}_t = [\theta_t, \dot{\theta}_t]^\top$, and the dynamics are nonlinear because of the term $\sin \theta_t$.

The objective here is to bring the rod to rest in the upright position $\theta = \pi$, using as little torque as possible. A simple stage cost is

$$c(\mathbf{x}_t, u_t) = \left\| \mathbf{x}_t - \begin{bmatrix} \pi \\ 0 \end{bmatrix} \right\|^2 + u_t^2.$$

When interpreting this objective globally, the angular error should be understood modulo 2π .

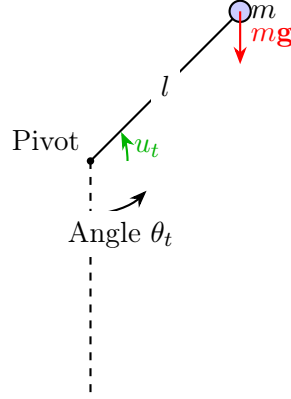


Figure 2.3: Pendulum swing-up illustration with marked angle θ_t , gravitational force mg , control torque u_t , mass m , and rod length l .

2.1.4 Autoregressive Language Models

A modern example of a dynamical system is an autoregressive language model. Let $\mathcal{V}$ be a finite vocabulary, and let $y_t \in \mathcal{V}$ denote the token observed at time t . The model maintains a hidden state h_t that summarizes the preceding context. For a recurrent neural network, one may take $h_t \in \mathbb{R}^d$ to be a fixed-dimensional latent vector. For a transformer, it is more natural to regard h_t as the model's internal cache or prefix representation associated with the sequence $y_{1:t}$; this state is typically very high-dimensional and is not directly observed.

A simple state-space description is

$$h_{t+1} = f_\theta(h_t, e(y_t)),$$

where $e(y_t)$ is the embedding of token y_t and f_θ is a learned nonlinear update map. From the hidden state, the model produces a probability distribution over the next token,

$$p_{t+1} = g_\theta(h_{t+1}) \in \Delta(\mathcal{V}),$$

where $\Delta(\mathcal{V})$ denotes the probability simplex over the vocabulary.

The objective in next-token prediction is to assign high probability to the realized next token y_{t+1} . A natural stage cost is therefore the log loss

$$c_t(h_t, y_{t+1}) = -\log p_{t+1}(y_{t+1}).$$

Equivalently, one may view training as minimizing the sum of these stage costs over time.

This example illustrates several features that are common in modern machine learning systems. The hidden state is extremely large and latent, the observations are discrete symbols rather than physical measurements, and the dynamics are highly nonlinear. At generation time, the model recursively predicts a distribution over the next token, chooses or samples a token from that distribution, appends it to the context, and repeats the process.

2.1.5 Epidemiological Models

The COVID-19 pandemic renewed public interest in policy interventions for controlling the spread of infectious diseases. A basic compartmental model is the SIR model. Let S_t , I_t , and R_t denote the fractions of the population that are susceptible, infected, and removed (recovered or immune), respectively, so that $S_t + I_t + R_t = 1$. Let $u_t \geq 0$ denote the vaccination effort at time t , and let $\alpha \in [0, 1]$ denote vaccine efficacy. A discrete-time controlled SIR model is

$$\begin{aligned} S_{t+1} &= S_t - \beta S_t I_t - \alpha u_t, \\ I_{t+1} &= I_t + \beta S_t I_t - \gamma I_t, \\ R_{t+1} &= R_t + \gamma I_t + \alpha u_t. \end{aligned}$$

Here, β measures transmissibility, γ is the recovery rate, and αu_t is the fraction of susceptible individuals effectively moved into the removed/immune compartment by vaccination. Under these dynamics, the identity $S_t + I_t + R_t = 1$ is preserved whenever it holds initially.

A natural control objective is to keep infections small while also limiting intervention cost, for example through the stage cost

$$c_t(S_t, I_t, R_t, u_t) = I_t^2 + \lambda u_t^2,$$

for some tradeoff parameter $\lambda > 0$, subject to $u_t \geq 0$ and $\alpha u_t \leq S_t$.

These examples span hidden-state and fully observed systems, linear and nonlinear dynamics, and a variety of natural control objectives. We next abstract away from the application details and formalize several general solution concepts.

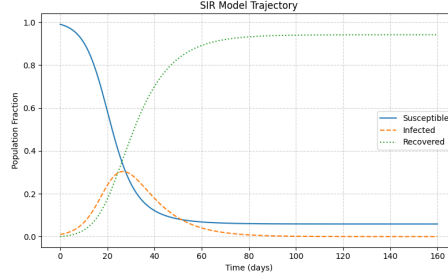


Figure 2.4: Typical trajectory of a continuous-time SIR model without vaccination, with parameters $\beta = 0.3$ and $\gamma = 0.1$.

2.2 Solution Concepts for Dynamical Systems

Before considering how to control a dynamical system, we should decide what success means. In this section, we discuss several solution concepts, including equilibrium, stability, stabilizability, and controllability.

Given a dynamical system $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t)$, its deterministic counterpart

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) = f(\mathbf{x}_t, \mathbf{u}_t, \mathbf{0})$$

models the evolution of the system in the absence of perturbations. The autonomous system

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t) = f(\mathbf{x}_t, \mathbf{0}, \mathbf{0})$$

describes the state transitions in the absence of both control inputs and perturbations. In both cases, we use the same symbol f when the omitted arguments are understood to be zero.

2.2.1 Equilibrium of Dynamical Systems

An example of an equilibrium is a ball sitting at rest at the bottom of a pit. If the ball is moved slightly, it returns to the bottom of the pit. Thus this equilibrium is both stable and attractive.

If a ball is at rest on the top of a mountain, then pushing it in any direction causes it to roll away. This is an equilibrium that is unstable. We now formalize these notions.

Definition 2.1. A point $\mathbf{x}_e$ is an **equilibrium point** of an autonomous dynamical system $\mathbf{x}_{t+1} = f(\mathbf{x}_t)$ if and only if $f(\mathbf{x}_e) = \mathbf{x}_e$.

The equilibrium $\mathbf{x}_e$ is **stable** if for every $\varepsilon > 0$ there exists $\delta > 0$ such that

$$\|\mathbf{x}_0 - \mathbf{x}_e\| < \delta \implies \|\mathbf{x}_t - \mathbf{x}_e\| < \varepsilon \quad \text{for all } t \geq 0.$$

The equilibrium $\mathbf{x}_e$ is **attractive** if there exists $\delta > 0$ such that

$$\|\mathbf{x}_0 - \mathbf{x}_e\| < \delta \implies \lim_{t \rightarrow \infty} \mathbf{x}_t = \mathbf{x}_e.$$

The equilibrium $\mathbf{x}_e$ is **asymptotically stable** if it is both stable and attractive. If the attraction property holds for every initial state in the state space, we say that the equilibrium is **globally attractive**; if it is also stable, we say that it is **globally asymptotically stable**.

Another example is given by a linear transformation in Euclidean space. Consider the system

$$\mathbf{x}_{t+1} = A\mathbf{x}_t.$$

Clearly $\mathbf{0}$ is an equilibrium point, since $A\mathbf{0} = \mathbf{0}$ for every matrix A . If every eigenvalue of A has modulus strictly less than one, then $A^t\mathbf{x}_0 \rightarrow \mathbf{0}$ for every initial state $\mathbf{x}_0$. Thus, $\mathbf{0}$ is a globally asymptotically stable equilibrium.

In general, determining or even describing equilibrium sets and long-run behavior can be difficult, as the examples later in this chapter illustrate.

2.2.2 Stabilizability and Controllability

Two fundamental properties of dynamical systems are stabilizability and controllability.

Definition 2.2. A dynamical system is **asymptotically stabilizable** if there exists a control policy π such that, for every initial state, the deterministic closed-loop trajectory satisfies

$$\lim_{t \rightarrow \infty} \mathbf{x}_t^\pi = \mathbf{0}.$$

Definition 2.3. A dynamical system is **controllable** if, for every initial state $\mathbf{x}_0$ and every target state $\mathbf{x}^*$, there exist a finite horizon T and controls $\mathbf{u}_0, \dots, \mathbf{u}_{T-1}$ such that the deterministic counterpart of the system satisfies $\mathbf{x}_T = \mathbf{x}^*$.

Controllability implies finite-time reachability of the zero state. If the zero state is an equilibrium that can be maintained once reached, then controllability also implies asymptotic stabilizability.

Some texts reserve the word *stabilizable* for a weaker algebraic property of linear systems. Here we use the more direct trajectory-based notion above, since it is convenient for our purposes.

In the next subsection, we introduce a more quantitative robust-boundedness notion that is useful in nonstochastic control.

2.2.3 Quantitative Definitions of Stabilizability

The notion below is stronger in one sense and weaker in another: it allows adversarial bounded disturbances, but it asks only that the state remain uniformly bounded rather than converge to zero. For linear systems, related ideas include robust invariance and input-to-state stability.

Definition 2.4. A control policy π for a dynamical system f is γ -**stabilizing** if the following holds. For every initial state satisfying $\|\mathbf{x}_0^\pi\| \leq 1$ and every disturbance sequence $\mathbf{w}_0, \dots, \mathbf{w}_{T-1}$ such that $\|\mathbf{w}_t\| \leq 1$ for all $t \in \{0, \dots, T-1\}$, let $\mathbf{x}_t^\pi$ be the trajectory generated by policy π :

$$\mathbf{x}_{t+1}^\pi = f(\mathbf{x}_t^\pi, \pi(\mathbf{x}_t^\pi), \mathbf{w}_t).$$

Then

$$\|\mathbf{x}_t^\pi\| \leq \gamma \quad \text{for all } t = 0, 1, \dots, T.$$

A dynamical system is said to be γ -**stabilizable** if and only if it admits a γ -stabilizing control policy.

2.3 Complexity of Dynamical Systems and Hardness of Stabilizability

Before formally proving the computational hardness of stabilizability, we begin with two examples that illustrate how complicated invariant sets and attractors can become even for simple dynamical rules.

Consider the complex quadratic map

$$z_{t+1} = f_c(z_t) = z_t^2 + c.$$

Its *filled Julia set* is

$$K_c = \left\{ z \in \mathbb{C} : \sup_{t \geq 0} |f_c^t(z)| < \infty \right\},$$

where $f_c^t(z)$ denotes t successive applications of the function f_c . The *Julia set* is the boundary $J_c = \partial K_c$. Depending on the parameter c , these sets

can have extremely intricate geometry, and there are strong negative results on the computability of Julia sets for certain values of c ; see, for example, [BY08]. A related parameter-space object is the Mandelbrot set, shown in Figure 2.5.

Another compelling visual illustration of complexity in dynamical systems is the Lorenz system, governed by the continuous-time equations

$$\begin{aligned}\dot{x} &= \sigma(y - x), \\ \dot{y} &= x(\rho - z) - y, \\ \dot{z} &= xy - \beta z.\end{aligned}$$

For classical parameter values, trajectories are attracted to a complicated invariant set known as the Lorenz attractor, depicted in Figure 2.6. For background on this attractor, see [Str14, Tuc99, Lor63]. These examples show that even describing the long-run behavior of a seemingly simple dynamical system can be a formidable task.

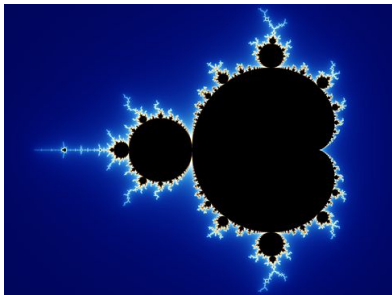


Figure 2.5: Mandelbrot set.

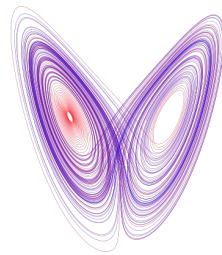


Figure 2.6: Lorenz attractor. Visualization by D Schwen, licensed under CC-BY-2.5.

2.3.1 Computational Hardness of Stabilizability

Consider the following decision problem: given a dynamical system, can we determine whether it is asymptotically stabilizable, that is, whether there exists a control policy that drives the system to the zero state from every initial condition? We now show that this decision problem is computationally hard.

Theorem 2.5. *For the family of discrete-time polynomial dynamical systems with integer coefficients, deciding asymptotic stabilizability is NP-hard.*

Proof. We reduce from 3SAT. Let

$$F(x_{1:n}) = c_1 \wedge c_2 \wedge \cdots \wedge c_m$$

be a Boolean formula in conjunctive normal form with three literals per clause. Introduce real variables $u_1, \dots, u_n$. For each literal ℓ , define

$$r(\ell) = \begin{cases} 1 - u_i, & \text{if } \ell = x_i, \\ u_i, & \text{if } \ell = \neg x_i. \end{cases}$$

For a clause $c = \ell_1 \vee \ell_2 \vee \ell_3$, let

$$g_c(\mathbf{u}) = r(\ell_1) r(\ell_2) r(\ell_3).$$

On Boolean assignments $\mathbf{u} \in \{0, 1\}^n$, the quantity $g_c(\mathbf{u})$ equals 1 exactly when the clause c is false and 0 otherwise.

Now define the polynomial

$$p_F(\mathbf{u}) = \sum_{i=1}^n (u_i(1 - u_i))^2 + \sum_{j=1}^m g_{c_j}(\mathbf{u})^2.$$

Then $p_F(\mathbf{u}) = 0$ if and only if $\mathbf{u} \in \{0, 1\}^n$ encodes a satisfying assignment of F . Indeed, the first sum vanishes if and only if each $u_i \in \{0, 1\}$, and the second sum vanishes if and only if no clause is false. Hence deciding whether a degree-six polynomial with integer coefficients has a real zero is NP-hard.

Consider now the scalar dynamical system

$$x_{t+1} = p_F(\mathbf{u}_t).$$

If F is satisfiable, choose a satisfying assignment $\mathbf{u}^*$ and apply the constant control $\mathbf{u}_t \equiv \mathbf{u}^*$. Then $p_F(\mathbf{u}^*) = 0$, so $x_t = 0$ for all $t \geq 1$, regardless of the initial state. Therefore the system is asymptotically stabilizable.

Conversely, suppose F is unsatisfiable. Then $p_F(\mathbf{u}) > 0$ for all $\mathbf{u} \in \mathbb{R}^n$. Since p_F is continuous and

$$\sum_{i=1}^n (u_i(1 - u_i))^2 \rightarrow \infty \quad \text{as } \|\mathbf{u}\| \rightarrow \infty,$$

the function p_F attains a strictly positive global minimum $m > 0$. Hence, for every control sequence,

$$x_{t+1} = p_F(\mathbf{u}_t) \geq m \quad \text{for all } t \geq 0.$$

So the state cannot converge to 0. Therefore the system is asymptotically stabilizable if and only if F is satisfiable, proving NP-hardness. $\square$

2.3. COMPLEXITY OF DYNAMICAL SYSTEMS AND HARDNESS OF STABILIZABILITY 25

The proof already shows hardness for an especially simple family of polynomial control systems. Exercise 2.1 asks the reader to strengthen this result and show that deciding asymptotic stabilizability for polynomial systems of degree four is also NP-hard.

2.4 Bibliographic Remarks

The text [Ted20] offers several examples and characterizations of dynamical systems useful for robotic manipulation. A more mathematically involved introduction appears in [Str14].

[Ada20] discusses further aspects of simulations used in the early days of the COVID-19 pandemic. See [SZG⁺21] for examples of control algorithms used for noninvasive mechanical ventilation.

The Lorenz equations were introduced to model the phenomenon of atmospheric convection [Lor63]. For detailed information on the mathematical properties of the Lorenz attractor, see [Str14, Tuc99]. For discussion of computability questions for Julia sets and related sets, see [BY08].

Nonlinear dynamical systems admit many stability notions beyond those introduced here, including Lyapunov stability, local and global asymptotic stability, robust invariance, and input-to-state stability. Lyapunov's direct method shows that suitable potential functions can certify stability without requiring explicit trajectory formulas; see [Lya92, Kha15] for further discussion.

An excellent survey on the computational complexity of control is [BT00]. For a more recent and stronger result on stabilizability, see [Ahm12]. The NP-hardness theorem in this chapter appeared in [Ahm16]. The computational complexity of mathematical programming—both constrained and unconstrained—is commented upon in [Haz19].

2.5 Exercises

2.1. Strengthen Theorem 2.5 to show that deciding whether a dynamical system given by a polynomial of degree four is asymptotically stabilizable is NP-hard. Hint: Reduce from the NP-hard problem of MAX-2-SAT instead of 3-SAT.

2.2. Let $A \in \mathcal{R}^{d \times d}$ be a diagonalizable matrix. Consider the following dynamical system:

$$\mathbf{x}_{t+1} = \begin{cases} \frac{A\mathbf{x}_t}{\|\mathbf{x}_t\|}, & \mathbf{x}_t \neq \mathbf{0}, \\ \mathbf{0}, & \mathbf{x}_t = \mathbf{0}. \end{cases}$$

What are the equilibrium points $\mathbf{x} \in \mathcal{R}^d$ of the system? Which of them are stable, attractive, or asymptotically stable? Are any of them globally attractive? Your answer may depend on A .

2.3. Let $\alpha, \beta, \gamma > 0$. Does the SIR model, as described in this chapter, define an asymptotically stabilizable dynamical system for a natural fixed choice of target state, such as a disease-free equilibrium? What about controllability? We assume that only starting states with the property $S_0 + I_0 + R_0 = 1$ are allowed and that $S_t, I_t, R_t, u_t \geq 0$ and $\alpha u_t \leq S_t$ hold at all times.

2.4. Consider the double integrator system:

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + Bu_t + \mathbf{w}_t,$$

where

$$A = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{x}_t = \begin{bmatrix} x_t \\ v_t \end{bmatrix} \in \mathbb{R}^2, \quad u_t \in \mathbb{R}, \quad \Delta > 0.$$

Assume throughout this exercise that $\mathbf{w}_t \equiv \mathbf{0}$.

The goal is to ensure that the system reaches the desired state $[x^*, 0]^\top$ as $t \rightarrow \infty$ from any initial condition.

1. Prove that (A, B) is controllable. That is, show that for any initial state and any target state, there exists a finite sequence of inputs that transitions the system from the initial state to the target state.
2. Design a stabilizing state-feedback law

$$u_t = -K(\mathbf{x}_t - [x^* \ 0]^\top),$$

with $K = [k_1 \ k_2]$, which places all eigenvalues of $A - BK$ strictly within the unit circle. Find conditions on Δ, k_1, k_2 that ensure that all eigenvalues of $A - BK$ are real and lie strictly between -1 and 1 , thus guaranteeing $\mathbf{x}_t \rightarrow [x^*, 0]^\top$.

Chapter 3

Markov Decision Processes

The previous chapters described and motivated the problem of control in dynamical systems. We have seen that, in full generality, even simple questions about dynamical systems such as stabilizability and controllability can be computationally hard. In this chapter, we study a more structured model for sequential decision making for which computationally efficient methods are available.

This chapter focuses on *finite-state, finite-action, discounted, fully observed* Markov decision processes. In this setting, the state space and transition dynamics are explicitly represented, and this extra structure makes efficient planning algorithms possible. In addition, throughout this chapter we assume that the transition probabilities and rewards are known. When these quantities are unknown and must be inferred from interaction with the environment, one arrives at reinforcement learning in the narrower sense.

After introducing the model of Markov decision processes, we discuss dynamic programming and the accompanying characterization of optimal policies via the Bellman equation. We then present two algorithmic consequences: a linear-programming formulation and the value-iteration algorithm.

3.1 Reinforcement Learning

Reinforcement learning is a subfield of machine learning that studies learning through interaction with a reactive environment. This is in contrast to learning from labeled examples, as in supervised learning, or inference in a probabilistic model, as in Bayesian statistics. In reinforcement learning, we have an agent and an environment. The agent observes its current state in

the environment and takes an action; the environment then transitions to a new state and emits a reward.

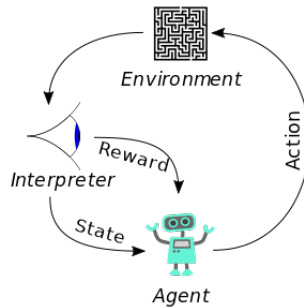


Figure 3.1: The agent–environment interaction loop in reinforcement learning.

Typically, the goal is to learn behavior that maximizes the total reward accumulated over time. This viewpoint has led to some notable successes: machines have learned how to play Atari games and how to defeat the strongest human players at Go.

A crucial modeling assumption is the following.

The Markovian assumption: Conditional on the current state and current action, the distribution of the next state and immediate reward is independent of the past.

The reader can imagine situations where such an assumption is reasonable. For example, in the inverted pendulum setting from Section 2.1.3, the assumption is natural: once the current angle and angular velocity are known, the future evolution under a chosen torque does not depend on how that state was reached.

Other problems call this assumption into question. In the medical ventilation problem, for instance, future treatment may depend on the medical history of the patient and not only on the present physiological variables. One possible remedy is to expand the state description so that it includes enough information to make the process Markovian. In the extreme, the state can contain the entire history, in which case the Markovian assumption becomes true by construction. Of course, such a representation may be computationally inefficient.

In this chapter we restrict attention to the fully observed setting, in which the state itself is available to the agent. When only partial observations

are available, the appropriate formal model is a partially observed Markov decision process (POMDP), which is beyond our present scope.

3.1.1 A Classical Example: Bubbo's World

Before diving into the formalism, we start with a classical example from the text of Russel and Norvig [RN02]. In this example, a robot starts in the bottom-left square, and its goal is to reach the terminal state with reward $+1$. The square marked -1 is also terminal, but undesirable. The block 'X' represents an obstacle through which the robot cannot travel.

			$+1$
	X		-1
Start			

Figure 3.2: A visual representation of Bubbo's world.

The robot can choose to move left, right, up, or down. However, the controls are noisy. With probability 0.8, the robot moves in the intended direction; with the remaining probability 0.2, it moves sideways, with probability 0.1 to each side.

Notice that the Markovian assumption clearly holds in this example. Once the robot's current square and chosen action are specified, the distribution of the next square and the immediate reward no longer depend on the earlier history.

A solution to this problem is a mapping from states to actions. Such a mapping is called a *policy*. The best policy depends not only on the transition structure but also on how rewards are assigned. To connect the example to the discounted infinite-horizon formalism used later in the chapter, it is convenient to model terminal states as absorbing states: entering the upper terminal state yields reward $+1$, entering the lower terminal state yields reward -1 , and thereafter the process remains in that terminal state with reward 0.

For example, if the reward for each non-terminal step is mildly negative, say -0.04 , then Bubbo prefers to steer toward the $+1$ terminal while avoiding the risk of falling into the -1 terminal, as in Figure 3.3.

→	→	→	+1
↑	X	↑	-1
↑	←	←	←

Figure 3.3: An optimal policy when the reward for each non-terminal step is -0.04 .

→	→	→	+1
↑	X	→	-1
→	→	→	↑

Figure 3.4: If the reward of each non-terminal step is changed to -2 , the optimal policy changes: the robot is now more concerned with ending the game quickly than with reaching the $+1$ terminal.

Conversely, if the step cost is very negative, then Bubbo prefers to terminate the episode as quickly as possible, even at the risk of ending in the -1 terminal. This leads to a different optimal policy, as in Figure 3.4.

We next describe a formal model that captures this and many other reinforcement-learning problems.

3.2 Markov Decision Processes

We now describe the mathematical model used in this chapter: finite discounted Markov decision processes. To ease the exposition, we begin with Markov chains, proceed to Markov reward processes, and then give the full Markov decision process definition.

3.2.1 Markov Chains

A Markov chain is a stochastic process that evolves without exogenous actions. It consists of a finite set of states together with a transition matrix that specifies the probability of moving from one state to another.

Formally, let S be a finite set of states. A Markov chain on S is specified by a row-stochastic matrix $P \in [0, 1]^{S \times S}$, where the entry P_{ij} is the probability of moving from state i to state j in one step. Figure 3.5 describes a Markov chain corresponding to a simple weather process with transition matrix

$$P = \begin{pmatrix} 0.9 & 0.1 & 0 & 0 \\ 0.5 & 0 & 0.4 & 0.1 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Denote the distribution over states at time t by $\mathbf{p}_t$. Then

$$\mathbf{p}_{t+1} = \mathbf{p}_t P = \mathbf{p}_0 P^{t+1},$$

where $\mathbf{p}_0$ is the initial state distribution.

A central object in the study of Markov chains is a stationary distribution.

Definition 3.1. *A distribution π over the state space S is a **stationary distribution** of a Markov chain with transition matrix P if and only if*

$$\pi = \pi P.$$

For finite Markov chains, a stationary distribution always exists. It need not be unique, however, and convergence from an arbitrary initial distribution need not occur without further assumptions.

Two standard structural properties are the following.

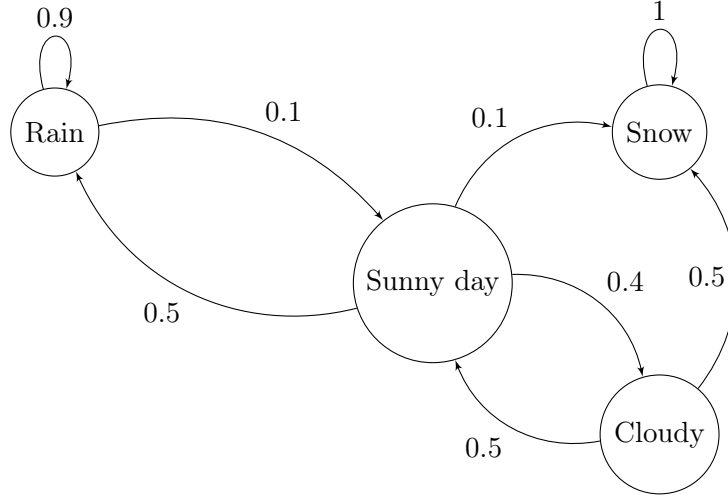


Figure 3.5: A graphical description of a Markov chain. Each state is the weather on a given day, and the transition probabilities describe the distribution of the next day's weather.

1. **Irreducibility.** A Markov chain is *irreducible* if and only if

$$\forall i, j \in S \exists t \geq 0 \text{ such that } (P^t)_{ij} > 0.$$

In words, every state is reachable from every other state.

2. **Aperiodicity.** A Markov chain is *aperiodic* if and only if every state has period one. The period of a state s_i is defined by

$$d(s_i) = \gcd \{ t \in \mathbb{N}_+ \mid (P^t)_{ii} > 0 \}.$$

The following finite-state ergodic theorem summarizes the role of these properties.

Theorem 3.2 (Ergodic theorem for finite Markov chains). *Let P be a finite Markov chain.*

1. P has at least one stationary distribution.
2. If P is irreducible, then the stationary distribution is unique.
3. If P is irreducible and aperiodic, then for every initial distribution $\mathbf{p}_0$,

$$\lim_{t \rightarrow \infty} \mathbf{p}_0 P^t = \boldsymbol{\pi},$$

where $\boldsymbol{\pi}$ is the unique stationary distribution.

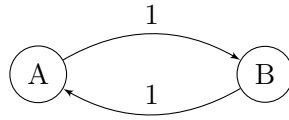


Figure 3.6: This two-state Markov chain is irreducible but periodic with period 2. It has a unique stationary distribution, namely $\pi = (\frac{1}{2}, \frac{1}{2})$, but the distribution of the chain does not converge to π from every initial state because the chain alternates deterministically between the two states.

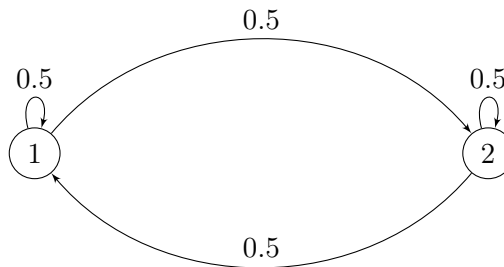


Figure 3.7: Adding a self-loop to every node makes the period equal to one. This chain is irreducible and aperiodic, so its stationary distribution exists, is unique, and equals $\pi = (\frac{1}{2}, \frac{1}{2})$.

Its proof is a beautiful application of spectral graph theory. Exercise 3.1 guides the reader through one standard proof of the convergence statement.

An important special case is the reversible Markov chain.

Definition 3.3. A Markov chain P is said to be **reversible with respect to a distribution π** if and only if, for every two states i, j ,

$$\pi_i P_{ij} = \pi_j P_{ji}.$$

A Markov chain is said to be **reversible** if there exists some distribution with respect to which it is reversible.

Exercise 3.4 asks the reader to prove that any distribution satisfying the reversibility equations is automatically stationary.

3.2.2 Markov Reward Processes

Markov reward processes add rewards to Markov chains. At each time step, the process transitions between states and emits a reward. For simplicity, we

assume rewards lie in the range $[0, 1]$. This normalization loses no essential generality and keeps the formulas uncluttered.

Definition 3.4. A *Markov reward process* is a tuple (S, P, R, γ) , where:

- S is a finite set of states.
- $P \in [0, 1]^{S \times S}$ is a row-stochastic transition matrix.
- $R \in [0, 1]^{S \times S}$ is a reward function, where $R(s, s')$ is the reward obtained when moving from state s to state s' .
- $\gamma \in [0, 1)$ is the discount factor.

We henceforth use the following notation.

1. $S_t \in S$ denotes the random state of the process at time t .
2. R_t denotes the random reward obtained when transitioning from S_t to S_{t+1} .
3. $\hat{R} \in \mathbb{R}^S$ denotes the vector of expected one-step rewards:

$$\hat{R}(s) = \mathbf{E}[R_t \mid S_t = s].$$

4. G_t denotes the discounted return from time t onward:

$$G_t = \sum_{i=0}^{\infty} \gamma^i R_{t+i}.$$

Since $\gamma < 1$ and rewards are bounded, the random variable G_t is well defined.

A natural question is: what quantity should summarize the future behavior of a Markov reward process? For an ordinary Markov chain, one often studies long-run state frequencies via stationary distributions. Once rewards and discounting are introduced, however, the central quantity becomes the expected discounted return from a given starting state. This dependence on the starting state is unavoidable: for example, when $\gamma = 0$, the return reduces to the immediate reward, which can vary from state to state.

This motivates the central object in the theory of Markov reward processes: the value function. Because the process is time-homogeneous, the conditional expectation $\mathbf{E}[G_t \mid S_t = s]$ depends only on the state s , and not on the particular time index t .

Definition 3.5. *The **value function** of a Markov reward process maps states to expected discounted return:*

$$\mathbf{v}(s) = \mathbb{E}[G_t \mid S_t = s].$$

The value function measures how desirable each state is in terms of expected future reward. It satisfies the Bellman equation for a Markov reward process:

$$\mathbf{v}(s) = \hat{R}(s) + \gamma \sum_{s' \in S} P_{s,s'} \mathbf{v}(s').$$

To verify this identity, write

$$\begin{aligned} \mathbf{v}(s) &= \mathbb{E}[G_t \mid S_t = s] \\ &= \mathbb{E} \left[\sum_{i=0}^{\infty} \gamma^i R_{t+i} \mid S_t = s \right] \\ &= \mathbb{E} \left[R_t + \gamma \sum_{i=0}^{\infty} \gamma^i R_{t+1+i} \mid S_t = s \right] \\ &= \mathbb{E}[R_t + \gamma G_{t+1} \mid S_t = s] \\ &= \mathbb{E}[R_t + \gamma \mathbf{v}(S_{t+1}) \mid S_t = s] \\ &= \hat{R}(s) + \gamma \sum_{s' \in S} P_{s,s'} \mathbf{v}(s'). \end{aligned}$$

In matrix form, the Bellman equation is

$$\mathbf{v} = \hat{R} + \gamma P \mathbf{v}.$$

Equivalently, $\mathbf{v}$ is the unique solution to the linear system

$$(I - \gamma P) \mathbf{v} = \hat{R}.$$

Since $\gamma < 1$ and P is a stochastic matrix, the matrix $I - \gamma P$ is invertible, and this system has a unique solution.

Thus, for a known Markov reward process, the value function is the solution to a linear system and can be computed in polynomial time in the number of states.

3.2.3 Markov Decision Processes

The final layer of abstraction is to allow actions, or controls in the language of control theory. This gives the full Markov decision process model.

Definition 3.6. A *Markov decision process* is a tuple (S, P, R, A, γ) such that:

1. S is a finite set of states.
2. A is a finite set of possible actions.
3. P is a transition kernel, written as

$$P_{ss'}^a = \Pr[S_{t+1} = s' \mid S_t = s, A_t = a].$$

For each action $a \in A$, the matrix P^a is row-stochastic.

4. R is a reward function on state–action–next-state triples, where $R_{ss'a}$ is the reward for moving from s to s' using action a . We also write

$$R_{sa} = \sum_{s' \in S} P_{ss'}^a R_{ss'a}$$

for the corresponding expected immediate reward. As above, we assume rewards lie in $[0, 1]$.

5. $\gamma \in [0, 1)$ is the discount factor.

The central solution concept is a policy.

Definition 3.7. A *policy* is a rule for choosing actions as the process evolves. A policy may in principle depend on the entire history. A *stationary policy* is a policy that depends only on the current state, and is therefore a mapping

$$\pi : S \rightarrow \Delta(A), \quad \pi(a \mid s) = \Pr[A_t = a \mid S_t = s].$$

If each $\pi(\cdot \mid s)$ is a point mass, the policy is called *deterministic*.

In this chapter we consider infinite-horizon discounted return. Starting from a state s , the value of a policy π is

$$\mathbf{v}_\pi(s) = \mathbf{E}_{\pi, P} \left[\sum_{t=0}^{\infty} \gamma^t R_t \mid S_0 = s \right].$$

The optimal value function is defined pointwise by

$$\mathbf{v}^*(s) = \sup_{\pi} \mathbf{v}_\pi(s).$$

A fundamental theorem of discounted MDPs is the following.

Theorem 3.8. *For every finite discounted Markov decision process, there exists a deterministic stationary optimal policy π^* . Moreover, every optimal policy has the same value function $\mathbf{v}^*$.*

The existence of an optimal stationary policy is a consequence of the Markovian structure of the problem and the Bellman optimality equation, which we now discuss.

3.3 The Bellman Equation

A fundamental property of Markov decision processes is that they satisfy a recursive optimality principle. This principle is expressed by the Bellman equation and is the basis for dynamic programming.

For a fixed stationary policy π , the Bellman equation takes the form

$$\mathbf{v}_\pi(s) = \sum_{a \in A} \pi(a | s) \left[R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}_\pi(s') \right].$$

In matrix–vector notation this can be written as

$$\mathbf{v}_\pi = \mathbf{R}^\pi + \gamma P^\pi \mathbf{v}_\pi,$$

where

1. $\mathbf{R}^\pi$ is the expected reward vector induced by π , defined by

$$R^\pi(s) = \mathbf{E}_{a \sim \pi(\cdot | s)}[R_{sa}],$$

2. P^π is the transition matrix induced by π , defined by

$$P_{ss'}^\pi = \sum_{a \in A} P_{ss'}^a \pi(a | s).$$

Since $\gamma < 1$, the matrix $I - \gamma P^\pi$ is invertible, and therefore the Bellman equation for a fixed policy has a unique solution.

The Bellman equation becomes especially powerful when applied to the optimal value function.

This is the **Bellman optimality equation**. It says that the optimal value of a state is obtained by choosing the action with the largest one-step reward plus discounted continuation value.

$$\mathbf{v}^*(s) = \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}^*(s') \right\}$$

The proof is a one-step decomposition. Fix a state s . For any policy, the value obtained from s is bounded above by the best possible choice of first action plus the optimal continuation value thereafter, so

$$\mathbf{v}^*(s) \leq \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}^*(s') \right\}.$$

Conversely, choose an action a^* achieving the maximum on the right-hand side. For each successor state s' , choose a policy that is ε -optimal from s' . By first taking action a^* and then following the chosen continuation policy, one achieves a value at least

$$R_{sa^*} + \gamma \sum_{s' \in S} P_{ss'}^{a^*} \mathbf{v}^*(s'),$$

and the reverse inequality follows.

The Bellman optimality equation immediately implies the existence of a deterministic optimal policy. Indeed, define a policy π^* by choosing at each state any maximizing action,

$$\pi^*(s) \in \arg \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}^*(s') \right\}.$$

Then $\mathbf{v}^*$ satisfies

$$\mathbf{v}^*(s) = R_{s\pi^*(s)} + \gamma \sum_{s' \in S} P_{ss'}^{\pi^*(s)} \mathbf{v}^*(s') \quad \text{for all } s \in S,$$

which is exactly the Bellman equation for the fixed policy π^* . By uniqueness of the solution of the Bellman equation for a fixed policy, we obtain $\mathbf{v}_{\pi^*} = \mathbf{v}^*$, proving that π^* is optimal. This proves Theorem 3.8.

3.3.1 Linear Programming Formulation

The Bellman optimality equation characterizes v^* as a fixed point. An equivalent characterization views v^* as the smallest function satisfying the Bellman inequalities.

Consider the following linear program:

$$\begin{aligned} \min_{\mathbf{v} \in \mathbb{R}^{|S|}} \quad & \sum_{s \in S} \mathbf{v}(s) \\ \text{s.t.} \quad & \mathbf{v}(s) \geq r(s, a) + \gamma \sum_{s' \in S} P(s' | s, a) \mathbf{v}(s') \quad \forall s \in S, a \in A. \end{aligned}$$

This program has $|S|$ variables and $|S| |A|$ constraints, so it can be solved in polynomial time in the numbers of states and actions. The next theorem shows that its unique optimum is precisely the optimal value function.

Theorem 3.9. *The unique optimal solution of the above linear program is the optimal value function $\mathbf{v}^*$.*

Proof. First, $\mathbf{v}^*$ is feasible, since it satisfies the Bellman optimality equation and therefore all Bellman inequalities.

Now let $\mathbf{v}$ be any feasible solution. Then for all s ,

$$\mathbf{v}(s) \geq \max_a \left\{ r(s, a) + \gamma \sum_{s'} P(s' | s, a) \mathbf{v}(s') \right\} = (T\mathbf{v})(s),$$

where T is the Bellman optimality operator. Thus $\mathbf{v} \geq T\mathbf{v}$. Iterating,

$$\mathbf{v} \geq T\mathbf{v} \geq T^2\mathbf{v} \geq \dots$$

Since $T^k\mathbf{v} \rightarrow \mathbf{v}^*$ as $k \rightarrow \infty$, it follows that $\mathbf{v} \geq \mathbf{v}^*$ componentwise.

Therefore $\mathbf{v}^*$ is the smallest feasible vector, and hence minimizes any strictly increasing objective such as $\sum_s \mathbf{v}(s)$. Uniqueness follows. $\square$

3.4 The Value Iteration Algorithm

The Bellman optimality equation also leads to one of the most fundamental algorithms for discounted MDPs: value iteration. Starting from an initial estimate of the value function, the algorithm repeatedly applies the Bellman optimality update.

The next theorem gives the basic convergence guarantee.

Theorem 3.10. *Let $(\mathbf{v}_t)_{t \geq 0}$ be the sequence produced by Value Iteration with initialization $\mathbf{v}_0 = \mathbf{0}$. Then:*

1. $\mathbf{0} \leq \mathbf{v}_t \leq \mathbf{v}_{t+1} \leq \mathbf{v}^*$ entry-wise for every $t \geq 0$.

Algorithm 2 Value Iteration

Input: MDPSet $\mathbf{v}_0(s) = 0$ for all states s **for** $t = 0$ to $T - 1$ **do**

Update for all states,

$$\mathbf{v}_{t+1}(s) = \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}_t(s') \right\}$$

end forDefine a greedy policy with respect to $\mathbf{v}_T$ by

$$\pi_T(s) \in \arg \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}_T(s') \right\}$$

return $\mathbf{v}_T$ and π_T

2. If we define the residual vector $\mathbf{r}_t = \mathbf{v}^* - \mathbf{v}_t$, then

$$\|\mathbf{r}_{t+1}\|_\infty \leq \gamma \|\mathbf{r}_t\|_\infty \quad \text{for every } t \geq 0.$$

Consequently,

$$\|\mathbf{v}^* - \mathbf{v}_t\|_\infty \leq \frac{\gamma^t}{1 - \gamma}.$$

In particular, $\mathbf{v}_t \rightarrow \mathbf{v}^*$ geometrically fast.*Proof.* Monotonicity and the upper bound are left as an exercise. We now prove the geometric decay of the residual.

For every state we have by choice of the algorithm that

$$\mathbf{v}_{t+1}(s) = \max_a \left\{ R_{sa} + \gamma \sum_{s'} P_{ss'}^a \mathbf{v}_t(s') \right\} \geq R_{sa^*} + \gamma \sum_{s'} P_{ss'}^{a^*} \mathbf{v}_t(s'),$$

where $a^* = \pi^*(s)$ is the action of an optimal policy at state s . On the other hand, from Bellman optimality equation, we have that

$$\mathbf{v}^*(s) = R_{sa^*} + \gamma \sum_{s'} P_{ss'}^{a^*} \mathbf{v}^*(s').$$

Subtracting the two equations we get for all states s that

$$\begin{aligned}\mathbf{r}_{t+1}(s) &= \mathbf{v}^*(s) - \mathbf{v}_{t+1}(s) \\ &\leq \gamma \sum_{s'} P_{ss'}^{a^*} (\mathbf{v}^*(s') - \mathbf{v}_t(s')) \\ &= \gamma \sum_{s'} P_{ss'}^{a^*} \mathbf{r}_t(s').\end{aligned}$$

Or in vector form, $\mathbf{r}_{t+1} \leq \gamma P^{a^*} \mathbf{r}_t$. Furthermore, $\mathbf{r}_t \geq 0$ is entry-wise non-negative for all t . (See Exercise 3.5.) Therefore, it holds that

$$\|\mathbf{r}_{t+1}\|_\infty \leq \gamma \|P^{a^*} \mathbf{r}_t\|_\infty \leq \gamma \|\mathbf{r}_t\|_\infty,$$

where we use the fact that each row of P^{a^*} sums to one. Applying this repeatedly, we have

$$\|\mathbf{r}_{t+1}\|_\infty \leq \gamma^t \|\mathbf{r}_1\|_\infty.$$

Finally, because rewards lie in $[0, 1]$, the total discounted reward is at most $1 + \gamma + \gamma^2 + \dots = 1/(1 - \gamma)$, so

$$\|\mathbf{r}_0\|_\infty = \|\mathbf{v}^*\|_\infty \leq \frac{1}{1 - \gamma}.$$

Combining the two inequalities yields

$$\|\mathbf{v}^* - \mathbf{v}_t\|_\infty = \|\mathbf{r}_t\|_\infty \leq \frac{\gamma^t}{1 - \gamma}.$$

□

3.5 Bibliographic Remarks

The section on reinforcement learning in the text [RN02] introduces basic concepts and places them in the broader context of artificial intelligence. The classical textbook on reinforcement learning is [SB18], which develops the subject from both the algorithmic and conceptual perspectives.

Another excellent reference is the set of lecture notes [Sil15]. See [AJKS19] for an introduction to the theoretical aspects of reinforcement learning, including sample-complexity guarantees and reductions to supervised learning. The reader may also refer to [Ber19] for a broad survey of computational approaches to learning in unknown Markov decision processes.

Dynamic programming is at the heart of Markov decision processes. This viewpoint is emphasized in [Ber07], which discusses the Bellman equations and related algorithmic ideas at length. It is also a useful source for the finite-horizon and average-reward settings, which we have only mentioned briefly here.

Although tabular MDPs do not compactly represent stochastic linear dynamical systems, function-approximation schemes [JYWJ20, DKL⁺21, JKA⁺17, KKL⁺20] can successfully capture problems such as stochastic LQR.

3.6 Exercises

3.1. In this exercise, we give a guided proof of the convergence part of the ergodic theorem. Consider a finite irreducible and aperiodic Markov chain with transition matrix P .

A. Show that the all-ones vector is a right eigenvector of P with eigenvalue 1. Conclude that 1 is also an eigenvalue of $P^\top$.

B. Show that all eigenvalues of P have magnitude at most one. The irreducibility and aperiodicity of the Markov chain imply that the eigenvalue 1 of $P^\top$ is simple and that all other eigenvalues have magnitude strictly smaller than one. Assume this fact, and let $\mathbf{v}_1$ denote the corresponding left eigenvector normalized to sum to one.

C. Consider an arbitrary initial distribution $\mathbf{p}_0$, and the distribution after t iterations of the Markov chain, $\mathbf{p}_t = \mathbf{p}_0 P^t$. Prove that

$$\lim_{t \rightarrow \infty} \mathbf{p}_t = \mathbf{v}_1.$$

3.2. Prove the geometric convergence of the value-iteration algorithm for rewards that can be arbitrary bounded real numbers, as opposed to lying in the range $[0, 1]$.

3.3. Let $\mathbf{v}^*$ be the optimal value function of a discounted Markov decision process, and define a greedy policy by

$$\pi(s) \in \arg \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}^*(s') \right\}.$$

Prove that π is optimal.

3.4. Let Markov chain P be reversible with respect to a distribution π , i.e.

$$\forall i, j, \pi_i P_{ij} = \pi_j P_{ji}.$$

Prove that π is stationary for P .

3.5. Suppose Value Iteration is initialized at an arbitrary vector $\mathbf{v}_0$ satisfying

$$\mathbf{v}_0(s) \leq \max_{a \in A} \left\{ R_{sa} + \gamma \sum_{s' \in S} P_{ss'}^a \mathbf{v}_0(s') \right\} \quad \text{for all } s \in S.$$

Prove that the successive iterates $\mathbf{v}_t$ increase monotonically, that is, $\mathbf{v}_{t+1} \geq \mathbf{v}_t$ for all t , and that $\mathbf{v}_t \leq \mathbf{v}^*$.

Chapter 4

Linear Dynamical Systems

In Chapter 2, we observed that even simple questions about general dynamical systems are intractable. This motivates the restriction to a special class of dynamical systems for which we can design efficient algorithms with provable guarantees.

Our treatment in this chapter is restricted to linear dynamical systems (LDS): dynamical systems that evolve linearly according to the equation

$$\mathbf{x}_{t+1} = A_t \mathbf{x}_t + B_t \mathbf{u}_t + \mathbf{w}_t.$$

If the matrices $A_t = A$ and $B_t = B$ are fixed, we say that the dynamical system is linear time-invariant (LTI). We have seen examples of LTI systems in Chapter 2, namely the double integrator and the aircraft dynamics.

A more general class of linear dynamical systems is when the matrices A_t and B_t change over time. These are called linear time-varying (LTV)

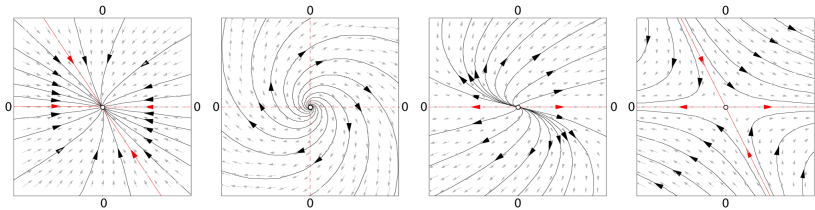


Figure 4.1: Four different deterministic linear time-invariant dynamical systems represented via their induced vector fields and a few chosen trajectories. Created by XaosBits using the Mathematica packages CurvesGraphics.nb by Gianluca Gorni and DrawGraphics.nb by David and Alice Park, licensed under CC-BY-2.5.

systems. The class of LTV systems can express more general dynamics via linearization, as we show next.

4.1 Approximating General Dynamics via LTV Systems

Consider any dynamics that can be expressed by a smooth function f . Such a system can be approximated by a time-varying linear system to an arbitrary accuracy determined by the time-step interval between samples.

To see this, first consider the one dimensional first-order Taylor expansion of a given dynamics function f , which is twice differentiable, around an anchor point $(\bar{x}_t, \bar{u}_t)$:

$$\begin{aligned} x_{t+1} &= f(x_t, u_t) + w_t \\ &= f(\bar{x}_t, \bar{u}_t) + \underbrace{\frac{\partial}{\partial x} f(x, u) \Big|_{\bar{x}_t, \bar{u}_t}}_{\stackrel{\text{def}}{=} A_t} (x_t - \bar{x}_t) + \underbrace{\frac{\partial}{\partial u} f(x, u) \Big|_{\bar{x}_t, \bar{u}_t}}_{\stackrel{\text{def}}{=} B_t} (u_t - \bar{u}_t) + w_t, \end{aligned}$$

where w_t is the approximation error, or remainder, which by the mean value theorem behaves as $w_t \sim (x_t - \bar{x}_t)^2 + (u_t - \bar{u}_t)^2$. Thus, this simplifies to

$$x_{t+1} = A_t x_t + B_t u_t + w_t.$$

An analogous approximation holds for the higher-dimensional case. The Taylor expansion is

$$f(\mathbf{x}, \mathbf{u}) = f(\bar{\mathbf{x}}, \bar{\mathbf{u}}) + J_{\mathbf{x}} f(\bar{\mathbf{x}}, \bar{\mathbf{u}})(\mathbf{x} - \bar{\mathbf{x}}) + J_{\mathbf{u}} f(\bar{\mathbf{x}}, \bar{\mathbf{u}})(\mathbf{u} - \bar{\mathbf{u}}) + w_t,$$

where $J_{\mathbf{x}} f(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ is the Jacobian of f with respect to variables $\bar{\mathbf{x}}$. Writing the matrices A_t, B_t for the Jacobians, we have

$$\mathbf{x}_{t+1} = A_t \mathbf{x}_t + B_t \mathbf{u}_t + \mathbf{w}_t.$$

This derivation explains why in practice the class of time-varying linear dynamical systems is very expressive, at least for smooth dynamical systems. This is particularly true with hidden states that can capture complicated dynamics while imposing enough structure for efficient algorithms.

As we shall see next, stabilizability and controllability can be checked efficiently for linear systems, in contrast to general dynamics.

4.2 Stabilizability of Linear Systems

Verifying stability of a system, and more so stabilizability, was shown to be intractable in our discussion of chapter 2. For linear dynamical systems, however, the picture is quite different: stability and stabilizability can be efficiently verified! Furthermore, the theoretical analysis of this fact is interesting and elegant.

For simplicity, in the rest of this section we consider time-invariant linear dynamical systems of the form

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t,$$

although much of our discussion can be generalized to LTV systems.

Recall that a system is stabilizable if, from every starting state, there is a sequence of controls that drives the system to zero. Determining whether a linear time-invariant system is stabilizable is determined by a particularly simple condition as follows. Henceforth, we denote by $\rho(A)$ the spectral radius of a matrix A , that is,

$$\rho(A) = \max\{|\lambda_1|, \dots, |\lambda_d|\},$$

the maximum modulus of all (possibly complex) eigenvalues. The stabilizability of a linear dynamical system can be characterized as follows.

Theorem 4.1. *A time-invariant LDS is stabilizable if and only if there exists a matrix K such that*

$$\rho(A + BK) < 1.$$

We therefore prove the sufficiency of this condition and leave the necessity as exercise 4.1. Both directions rely on the characterization of the spectral norm as follows:

Fact 4.2. *$\rho(A) < 1$ if and only if $\lim_{t \rightarrow \infty} A^t = 0$. Furthermore, if $\rho(A) > 1$, then for any norm, $\lim_{t \rightarrow \infty} \|A^t\| = \infty$.*

The proof of this fact is left as exercise 4.2. Based on this fact, we can now prove the sufficiency part of Theorem 4.1.

Theorem 4.1, sufficiency. Suppose that there exists a K such that $\rho(A + BK) < 1$. Then choose all controls to be $\mathbf{u}_t = K\mathbf{x}_t$. The evolution of the system without perturbation is given by

$$\mathbf{x}_t = (A + BK)\mathbf{x}_{t-1} = \dots = (A + BK)^t \mathbf{x}_0.$$

Thus, using the previous fact concerning the spectral radius, we have

$$\lim_{t \rightarrow \infty} \|\mathbf{x}_t\| = \lim_{t \rightarrow \infty} \|(A + BK)^t x_0\| = \|0\mathbf{x}_0\| = 0.$$

□

4.2.1 Efficient Verification of Stabilizability

The spectral radius of a matrix can be computed to an arbitrary accuracy in polynomial time and can be approximated by the power method. This implies that the stability of a given system can be efficiently verified.

To efficiently check if a given system is stabilizable, we make note of the following result due to Lyapunov. We use the notation $A \succeq 0$ to denote that the matrix A is positive semidefinite, meaning that it is symmetric and all its eigenvalues are nonnegative.

Theorem 4.3. $\rho(A) < 1$ holds if and only if there exists a matrix $P \in \mathbb{R}^{d_x \times d_x}$ such that $P \succeq I$ and $P \succ A^\top P A$.

Proof. Say there exists a matrix P such that $P \succeq I$ and $P \succ A^\top P A$, then to show that $\rho(A) < 1$, it suffices to prove that $\lim_{t \rightarrow \infty} A^t = 0$. To observe the latter, first note that there exists a positive constant $c < 1$ such that $cP \succeq A^\top P A$, implying that $c^t P \succeq (A^t)^\top P A^t \succeq (A^t)^\top A^t$ via repeated applications of the previous inequality. Therefore, we have $\lim_{t \rightarrow \infty} (A^t)^\top A^t = 0$, and consequently $\lim_{t \rightarrow \infty} A^t = 0$ since $\|A^\top\| \|A\| = \|A^\top A\|$.

For the other direction, given some matrix A with $\rho(A) < 1$, consider $P = \sum_{t=0}^{\infty} (A^t)^\top A^t$. Since $P = I + A^\top P A$, P satisfies the required conclusions. The fact that P as defined exists, that is, that it is finite, as long as $\rho(A) < 1$ can be verified via the Jordan normal form of A , which in particular asserts that $\|A^t\| \leq O(t^{d_x} \rho(A)^t)$. □

Using this observation, one can write a semidefinite mathematical program that checks if a given system is stabilizable, and if so produces a matrix K that guarantees a spectral radius smaller than one.

Theorem 4.4. There exists a matrix K such that $\rho(A + BK) < 1$ if and only if there exist matrices $P \in \mathcal{S}^{d_x \times d_x}$ and $L \in \mathbb{R}^{d_u \times d_x}$ such that

$$\begin{bmatrix} P & (AP + BL)^\top \\ AP + BL & P \end{bmatrix} \succ 0, \quad P \succeq I. \quad (4.1)$$

Furthermore, given such matrices P and L satisfying Equation 4.1, the matrix $K' = LP^{-1}$ satisfies $\rho(A + BK') < 1$.

Proof. We begin by observing the following result, using the Schur complement, regarding the semi-definiteness of block matrices.

Lemma 4.5. *Define a block matrix M as*

$$M = \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}. \quad (4.2)$$

Then, $M \succ 0$ if and only if $A \succ 0$ and $C - B^\top A^{-1} B \succ 0$.

Note that since the spectral radius purely involves the eigenvalues of a matrix, it is invariant under transposition, and we have $\rho((A + BK)^\top) = \rho(A + BK)$. Now using this fact and Theorem 4.3, we know that the stabilizability of a linear system is equivalent to the existence of matrices K and P such that $P \succeq I$ and $P \succ (A + BK)P(A + BK)^\top$. The above lemma allows us to reformulate this cubic constraint to a quadratic constraint. The stabilizability of a system is equivalent to existence of matrices K and P such that

$$\begin{bmatrix} P & ((A + BK)P)^\top \\ (A + BK)P & P \end{bmatrix} \succ 0, \quad P \succeq I.$$

Lastly, we reparameterize this constraint as a linear semidefinite constraint as demonstrated below, by choosing $L = KP$.

$$\begin{bmatrix} P & (AP + BL)^\top \\ AP + BL & P \end{bmatrix} \succ 0, \quad P \succeq I.$$

Notice that this is indeed a reformulation of the mathematical feasibility program, and not a relaxation, since the reparamertization $K = LP^{-1}$ is invertible. $\square$

4.3 Controllability of Linear Dynamical Systems

Recall that a system is controllable if and only if in the absence of noise and for every target state there is a sequence of controls that drives the system state to that target. Linear dynamical systems exhibit a particularly elegant characterization of controllability. First, we need to define an object of central importance for linear dynamical systems, the Kalman matrix.

Definition 4.6. *The Kalman controllability matrix of a linear dynamical system is given by*

$$K_r(A, B) = [B, AB, A^2B, \dots, A^{r-1}B] \in \mathbb{R}^{d_x \times r d_u}.$$

The controllability of a linear dynamical system can now be characterized using this matrix. Recall that the rank of a matrix is the maximal number of linearly independent columns (or rows).

Theorem 4.7. *A time-invariant linear dynamical system is controllable if and only if*

$$\text{rank } K_{d_x}(A, B) = d_x.$$

Proof. Consider the evolution of the noiseless linear dynamical system starting from the initial state $\mathbf{x}_0$. The state at time T is given by the expression

$$\begin{aligned} \mathbf{x}_T &= A\mathbf{x}_{T-1} + B\mathbf{u}_{T-1} \\ &= A^2\mathbf{x}_{T-2} + AB\mathbf{u}_{T-2} + B\mathbf{u}_{T-1} \\ &= \dots = \sum_{t=1}^T A^{t-1}B\mathbf{u}_{T-t} + A^T\mathbf{x}_0. \end{aligned}$$

Thus,

$$\mathbf{x}_T - A^T\mathbf{x}_0 = [B, AB, A^2B, \dots, A^{T-1}B] \begin{bmatrix} \mathbf{u}_{T-1} \\ \vdots \\ \mathbf{u}_0 \end{bmatrix}.$$

By the fundamental theory of linear algebra, this equation has a solution if and only if $K_T(A, B)$ has full rank. That is, for every initial state $\mathbf{x}_0$ and final state $\mathbf{x}_T$, there exists a set of control vectors $\{\mathbf{u}_t\}$ that satisfy the equation if and only if $K_T(A, B)$ has full rank. The rank of the Kalman matrix is upper bounded by the number of rows of K_T , which is d_x . Thus, the system is controllable if and only if the rank of K_T is d_x for all $T \geq d_x$.

Denote by $r_t = \text{rank}(K_t)$ the rank of the Kalman matrix after t iterations. Since the rank can never decrease with t , it remains to show that the system is controllable if and only if $r_t = r_{d_x}$ for all $t \geq d_x$.

We claim that once $r_t = r_{t+1}$, the rank never increases again by increasing t . Assuming $r_t = r_{t+1}$, we have

$$\text{span}[B, AB, \dots, A^t B] = \text{span}[B, AB, \dots, A^{t-1} B].$$

Since linear transformations retain containment, we can multiply by A and get

$$\text{span}[AB, \dots, A^{t+1} B] \subseteq \text{span}[AB, \dots, A^t B].$$

Adding B to both matrices does not affect the subset relationship.

$$\text{span}[B, AB, \dots, A^{t+1} B] \subseteq \text{span}[B, AB, \dots, A^t B]$$

Therefore, $r_{t+2} \leq r_{t+1}$. Since the rank cannot decrease through the addition of columns, we have $r_{t+2} = r_{t+1}$. Since the rank can grow at most d_x times, equation $r_{d_x} = r_T$ holds for all $T > d_x$ if and only if the system is controllable, and this concludes the theorem. $\square$

4.4 Quantitative Definitions

In the rest of this text our goal is to give efficient algorithms with finite time performance bounds for online control. These finite-time bounds naturally depend on how difficult it is for a given system to stabilize. Therefore, we require a quantitative definition for stabilizability, one that holds for both time-invariant and time-varying linear dynamical systems.

4.4.1 Stabilizability of Time-Invariant Systems

Consider a linear time-invariant system given by the transformations (A, B) . We have already remarked that this system is called stabilizable, according to the classical definition, if and only if there exists K such that $\rho(A + BK) < 1$ holds.

A stronger quantitative property can be obtained as follows, and in fact it is implied by BIBO stabilizability as defined in Definition 2.4. The proof of this fact is left as exercise 4.5.

Lemma 4.8. *Consider a linear controller K for the LTI system (A, B) . If there exist $\kappa > 0$ and $\delta \in (0, 1]$ such that the matrix $A + BK$ admits the following inequality for all t , then K is a γ -stabilizing controller for (A, B) .*

$$\|(A + BK)^t\| \leq \kappa(1 - \delta)^t, \quad \frac{\kappa}{\delta} \leq \gamma$$

Proof. Let $\tilde{K} = A + BK$. By unrolling the dynamics equation we have that

$$\begin{aligned} \mathbf{x}_{t+1} &= A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t = \tilde{K}\mathbf{x}_t + \mathbf{w}_t, \\ &= \sum_{i=0}^{t-1} \tilde{K}^i \mathbf{w}_{t-i} + \tilde{K}^{t+1} \mathbf{x}_0. \end{aligned}$$

Thus, as long as $\|\mathbf{w}_t\|, \|\mathbf{x}_0\| \leq 1$, we have

$$\begin{aligned} \|\mathbf{x}_{t+1}\| &\leq \sum_{i=0}^t \|\tilde{K}^i\| \|\mathbf{w}_{t-i}\| + \|\tilde{K}^{t+1}\| \|\mathbf{x}_0\|, \\ &\leq \kappa \sum_{i=0}^{\infty} (1 - \delta)^i \leq \frac{\kappa}{\delta} \leq \gamma. \end{aligned}$$

□

A linear time-invariant system is thus γ -stabilizable if we can find a matrix K with the above properties.

4.4.2 Stabilizability of Time-Varying Systems

The γ -stabilizability condition, as opposed to classical stabilizability, is particularly important to generalize to time-varying systems, since the spectral radius of the product is **not** bounded by the product of spectral radii, see exercise 4.3 at the end of this chapter.

We now define a sufficient condition for γ -stabilizability. Consider a time-varying linear dynamical system given by the transformations $\{A_t, B_t\}$.

Lemma 4.9. *Consider a sequence of linear controllers $K_{1:T}$ for the time-varying linear dynamical system $\{A_{1:T}, B_{1:T}\}$. If the matrices $\tilde{K}_t = A_t + B_t K_t$ satisfy the following inequality, then this linear controller sequence is γ -stabilizing.*

$$\forall t_1, t_2 . \left\| \prod_{t=t_1}^{t_2} \tilde{K}_t \right\| \leq \kappa(1 - \delta)^{t_1 - t_2}, \quad \frac{\kappa}{\delta} \leq \gamma.$$

Proof. Notice that under the condition of the lemma, by unrolling the dynamics equation we have that

$$\begin{aligned} \|\mathbf{x}_{t+1}\| &= \|A_t \mathbf{x}_t + B_t \mathbf{u}_t + \mathbf{w}_t\| = \|\tilde{K}_t \mathbf{x}_t + \mathbf{w}_t\|, \\ &= \left\| \sum_{i=0}^{t-1} \left(\prod_{j=1}^i \tilde{K}_{t-j} \right) \mathbf{w}_{t-i} + \left(\prod_{j=1}^t \tilde{K}_{t-j} \right) \mathbf{x}_0 \right\|, \\ &\leq \kappa \sum_{i=0}^{\infty} e^{-\delta i} \leq \frac{\kappa}{\delta} \leq \gamma. \end{aligned}$$

□

4.4.3 Controllability of Time-Invariant Systems

Consider the following one-dimensional dynamical system for some small $\varepsilon > 0$:

$$x_{t+1} = x_t + \varepsilon u_t.$$

Clearly, such a system is nominally controllable, as is indeed any one-dimensional system with nonzero B . However, reaching a state $x = 1$ from an initial

$x_0 = 0$ in a constant number of time steps requires a large control input u_t of magnitude at least $1/\varepsilon$. Thus, such a system may be said to be barely controllable. This difficulty of controllability is captured in the definition below, which requires not only that $K(A, B)$ is a full row-rank, but also that such a statement is robust to small changes in A, B .

Definition 4.10. *A time-invariant linear dynamical system (A, B) is (k, κ) -strongly controllable if and only if*

$$\text{rank } K_k(A, B) = d_x, \text{ and } \|(K_k(A, B)K_k(A, B)^\top)^{-1}\| \leq \kappa.$$

In the above definition, in addition to the full row-rank constraint, the spectral norm of the inverse of the Gram matrix of $K(A, B)$ is bounded by κ . For example, in the simple one-dimensional system we presented above, with $k = 1$, the matrix $K(A, B)$ is ε , and thus $\kappa = \frac{1}{\varepsilon^2}$, which indeed captures the complexity of controlling the system.

4.5 Bibliographic Remarks

The idea of reducing a nonlinear dynamical system to a time varying linear dynamical system had been independently discovered in disparate fields. In the context of control, using the linearization to characterize local stability properties of non-linear systems was pioneered by Lyapunov [Lya92], and is called the Lyapunov direct method; see, e.g. [SL91].

The asymptotic notions of stability, stabilizability and controllability discussed in this chapter constitute a core part of theory of linear control, and are found in most texts on the subject (e.g. [Ste94, Hes18]).

The efficient verification of stabilizability via a semidefinite program and the recovery of a stable controller from the solution of the latter appears in [Ahm16]. On the other hand, it is known that even imposition of norm constraints on a plausibly stable controller makes this problem computationally hard [BT97, BT00, BT99]. Somewhat distinct but related problems of stabilizability with even a 1-dimensional linear constraint are open to the date.

The notion of γ -stabilizability has a precursor in the notion of strong stability introduced in [CHK⁺18]. An extension of this strong stability concept to time-varying linear dynamics was established in [CKM19]. The definition presented in this chapter has weaker requirements than those in the listed articles, allowing us to work with and guarantee regret later with respect to a larger class of comparator policies.

4.6 Exercises

4.1. * Prove that the existence of a matrix K such that $\rho(A + BK) < 1$ is a necessary condition for a linear dynamical system to be stabilizable. Hint: this exercise requires the optimal solution to LQR in the next chapter.

4.2. Prove that $\rho(A) < 1$ if and only if $\lim_{t \rightarrow \infty} A^t = 0$. Furthermore, if $\rho(A) > 1$, then for any norm, $\lim_{t \rightarrow \infty} \|A^t\| = \infty$. Finally, prove that $\rho(A) = \lim_{t \rightarrow \infty} \|A^t\|^{1/t}$.

4.3. Prove that the spectral radius of a product of matrices is not necessarily bounded by the product of the spectral radii of the individual matrices. That is, show that in general,

$$\rho\left(\prod_{i=1}^n A_i\right) \not\leq \prod_{i=1}^n \rho(A_i).$$

4.4. Prove that a stochastic linear dynamical system with Gaussian perturbations

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + \mathbf{w}_t, \quad \mathbf{w}_t \sim \mathcal{N}(0, I),$$

constitutes a reversible Markov chain if and only if A is symmetric, i.e. $A = A^\top$.

Hint: One way to do this exercise is to explicitly calculate the stationary distribution of an LDS and then to verify reversibility as stated in its definition. Exercise ?? provides an alternative without the need to derive the stationary distribution.

4.5. This exercise explores the consequences of γ -BIBO stability (Definition 2.4) for linear systems. Although we have already seen sufficient conditions to ensure BIBO stability, we ask for the necessity of such conditions.

A. For a time-invariant linear system (A, B) equipped with a linear policy K , prove that BIBO stability indeed implies $\rho(A + BK) < 1$. In fact, comment on how large $\rho(A + BK)$ can be.

B. For a time-invariant BIBO-stable linear system, prove that there exist $\kappa > 0$ and $\delta \in (0, 1]$ such that, for all t , $\|(A + BK)^t\| \leq \kappa(1 - \delta)^t$. In particular, this proves that the conditions listed in Lemma 4.8 are necessary, in addition to being sufficient.

C. Construct a time-varying linear system that is BIBO stable but does not have the exponential decay properties required in Lemma 4.9.

4.6. Prove that the bound of one on the operator norm of the matrix A is a sufficient but not necessary condition for stabilizability.

Chapter 5

Optimal Control of Linear Dynamical Systems

In the chapters thus far, we have studied more basic notions than control, namely solution concepts of dynamical systems such as stabilizability and controllability. Even for these more basic problems, we have seen that they are computationally intractable in general.

This motivated us to look into simpler dynamics, namely linear dynamical systems, for which we can at least answer basic questions about stabilizability and controllability. In this chapter, we consider how we can control linear dynamical systems optimally.

The answer is positive in a strong sense. An archetypal problem in optimal control theory is the control of linear dynamical systems with quadratic cost functions. This setting is called the Linear Quadratic Regulator which we define and study in this chapter. The resulting efficient and practical algorithm is one of the most widely used subroutines in control.

5.1 The Linear Quadratic Regulator

The linear quadratic regulator (LQR) problem is that of optimal control of a linear dynamical system with known and fixed quadratic costs. For simplicity, we mostly consider the time-invariant version, where the linear dynamical system is fixed, and the costs are also fixed quadratic, as given formally below:

Definition 5.1. *The (time-invariant) Linear Quadratic Regulator (LQR) input is a linear dynamical system that evolves according to*

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t,$$

where $\mathbf{w}_t \sim \mathcal{N}(0, \sigma^2)$ is i.i.d. Gaussian, and convex quadratic cost functions are specified by matrices Q and R .

The goal is to generate a sequence of controls $\mathbf{u}_t$ that minimizes the expected cost, taken over the randomness of the perturbations, given by quadratic loss functions,

$$\min_{\mathbf{u}(\mathbf{x})} \mathbb{E} \left[\sum_{t=1}^T \mathbf{x}_t^\top Q \mathbf{x}_t + \mathbf{u}_t^\top R \mathbf{u}_t \right].$$

To remind the reader, the notation we use is as follows.

1. $\mathbf{x}_t \in \mathbb{R}^{d_x}$ is the state.
2. $\mathbf{u}_t \in \mathbb{R}^{d_u}$ is the control input.
3. $\mathbf{w}_t \in \mathbb{R}^{d_w}$ is the perturbation sequence drawn i.i.d. from normal distributions, or any other zero-mean distribution as long as the draws are independent of states or controls.
4. $A \in \mathbb{R}^{d_x \times d_x}$, $B \in \mathbb{R}^{d_x \times d_u}$ are the system matrices.
5. $Q \in \mathbb{R}^{d_x \times d_x}$, $R \in \mathbb{R}^{d_u \times d_u}$ are positive definite matrices. Thus, the cost functions are convex.

This framework is very useful, since many physical problems, such as the centrifugal governor and the ventilator control problem, can be approximated by a linear dynamical system. In addition, quadratic convex cost functions are natural in many physical applications.

The LQR is a special case of a Markov decision process which we have described in Chapter 3, since the transitions are stochastic. However, we cannot naively apply value iteration, policy iteration, or linear programming since there are infinitely many states and actions.

We proceed to describe a particularly elegant solution to the LQR problem that makes use of its special structure.

5.2 Optimal Solution of the LQR

We know from Chapter 3 that the LQR is a special case of a finite-time Markov decision process and hence admits an optimal policy. However, the

optimal policy is in general a mapping from states to actions, both of which are infinite Euclidean spaces in our case.

However, it can be shown that the optimal policy for an LQR is a **linear policy**, that is, a linear mapping from $\mathbb{R}^{d_x}$ to $\mathbb{R}^{d_u}$! This astonishing fact, together with a remarkable property of the value function, is formally given in the following theorem. We state the theorem for a finite number of iterations and for a linear time-invariant system. Generalizing this to a time-varying linear system is left as exercise 5.1.

Theorem 5.2. *For the LQR problem with finite $T > 0$ iterations, there exist positive semi-definite matrices S_t , K_t and scalars c_t , such that the optimal value function and optimal policy satisfy:*

$$\mathbf{v}_t^*(\mathbf{x}) = \mathbf{x}^\top S_t \mathbf{x} + c_t, \quad \mathbf{u}_t^*(\mathbf{x}) = K_t \mathbf{x}.$$

Proof. Recall that the Bellman Optimality equation for a general MDP is written as

$$\mathbf{v}_t^*(s) = \max_a \left\{ R_{sa} + \sum_{s'} P_{ss'}^a \mathbf{v}_{t+1}^*(s') \right\}.$$

We henceforth write the special case applicable to LQR, and prove the theorem by backward induction on t .

Base Case: For $t = T$, we have

$$\mathbf{v}_T^*(\mathbf{x}) = \min_{\mathbf{u}} \{ \mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} \}.$$

Since $R \succ 0$, a minimum occurs at $\mathbf{u} = 0$. Hence, $S_T = Q \succ 0$, $K_T = 0$ and $c_T = 0$.

Inductive Step: Assume for some $t > 1$ that $\mathbf{u}_t^*(\mathbf{x}) = K_t \mathbf{x}$ and $\mathbf{v}_t^*(\mathbf{x}) = \mathbf{x}^\top S_t \mathbf{x} + c_t$. Then we have

$$\begin{aligned} \mathbf{v}_{t-1}^*(\mathbf{x}) &= \min_{\mathbf{u}} \{ \mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} + \mathbf{E}_{\mathbf{w}} \mathbf{v}_t^*(A\mathbf{x} + B\mathbf{u} + \mathbf{w}) \} \\ &= \min_{\mathbf{u}} \{ \mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} + \mathbf{E}_{\mathbf{w}} [(A\mathbf{x} + B\mathbf{u} + \mathbf{w})^\top S_t (A\mathbf{x} + B\mathbf{u} + \mathbf{w})] \} + c_t \\ &= \min_{\mathbf{u}} \{ \mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} + (A\mathbf{x} + B\mathbf{u})^\top S_t (A\mathbf{x} + B\mathbf{u}) + \mathbf{E}_{\mathbf{w}} [\mathbf{w}^\top S_t \mathbf{w}] \} + c_t, \end{aligned}$$

where in the last equality we used the fact that $\mathbf{E}[\mathbf{w}] = 0$, and that the perturbation is independent of the state and control. We henceforth denote the second moment of the noise by $\mathbf{E}[\mathbf{w}\mathbf{w}^\top] = \sigma^2 I$. This implies that for any matrix M which is independent of $\mathbf{w}$, we have $\mathbf{E}[\mathbf{w}^\top M \mathbf{w}] = \mathbf{E}[M \bullet \mathbf{w}\mathbf{w}^\top] = \sigma^2 \cdot M \bullet I = \sigma^2 \text{Tr}(M)$.

Since R and S_t are positive semidefinite, the above expression is convex as a function of $\mathbf{u}$. At the global minimum the gradient vanishes, and hence

$$\nabla_{\mathbf{u}} \left[\mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} + (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u})^\top S_t (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}) + \sigma^2 \text{Tr}(S_t) \right] = 0.$$

Since Q, R, S_t are all positive semi-definite and symmetric, this gives us

$$2R\mathbf{u} + 2B^\top S_t \mathbf{A}\mathbf{x} + 2B^\top S_t \mathbf{B}\mathbf{u} = 0 \implies (R + B^\top S_t B)\mathbf{u} = -B^\top S_t \mathbf{A}\mathbf{x}.$$

Hence, we have that

$$\mathbf{u}_{t-1}(\mathbf{x}) = K_{t-1}\mathbf{x}, \quad K_{t-1} = -(R + B^\top S_t B)^{-1}(B^\top S_t A).$$

Notice that the matrix $R + B^\top S_t B$ is full rank since $R \succ 0$. Plugging this in the expression for the value function, we get

$$\begin{aligned} \mathbf{v}_{t-1}^*(\mathbf{x}) &= \mathbf{x}^\top Q \mathbf{x} + \mathbf{x}^\top K_{t-1}^\top R K_{t-1} \mathbf{x} + \mathbf{x}^\top (A - B K_{t-1})^\top S_t (A - B K_{t-1}) \mathbf{x} + \sigma^2 \text{Tr}(S_t) + c_t \\ &= \mathbf{x}^\top S_{t-1} \mathbf{x} + c_{t-1} \end{aligned}$$

where $S_{t-1} = Q + K_{t-1}^\top R K_{t-1} + (A - B K_{t-1})^\top S_t (A - B K_{t-1})$ is positive definite, and $c_{t-1} = c_t + \sigma^2 \text{Tr}(S_t)$.

□

5.3 Infinite Horizon LQR

Although infinite horizon problems can never appear in the real world, in practice the horizon length may be unknown. Although there are techniques to iteratively update the horizon length with geometrically increasing guesstimates, these are slow and impractical.

An alternative is to solve the LQR for the infinite horizon case, which results in a very elegant solution. Not only that, this turns out to be a practical solution to which the finite-horizon solution converges quickly.

The infinite horizon LQR problem can be written as follows:

$$\begin{aligned} \min_{\mathbf{u}(\mathbf{x})} \lim_{T \rightarrow \infty} \frac{1}{T} \mathbb{E} \left[\sum_{t=1}^T \mathbf{x}_t^\top Q \mathbf{x}_t + \mathbf{u}_t^\top R \mathbf{u}_t \right] \\ \text{s.t. } \mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t + \mathbf{w}_t, \end{aligned}$$

where similar to the finite horizon case, we assume that the perturbations $\mathbf{w}_t$ are i.i.d. zero mean and have bounded variance. We henceforth assume that the perturbations are all zero for simplicity, and adding zero-mean bounded-variance perturbations is left as exercise 5.2.

Without perturbations, the Bellman optimality equation for the total cost becomes

$$\mathbf{v}^*(\mathbf{x}) = \min_{\mathbf{u}} \left\{ \mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} + \mathbf{v}^*(A\mathbf{x} + B\mathbf{u}) \right\}.$$

Assuming that the value function is quadratic, we have:

$$\mathbf{x}^\top S \mathbf{x} = \min_{\mathbf{u}} \left\{ \mathbf{x}^\top Q \mathbf{x} + \mathbf{u}^\top R \mathbf{u} + (A\mathbf{x} + B\mathbf{u})^\top S (A\mathbf{x} + B\mathbf{u}) \right\}.$$

Similarly to the finite-time case, we have the following result.

Lemma 5.3. *The optimal policy is given by a transformation $\mathbf{u}^*(\mathbf{x}) = K\mathbf{x}$.*

Proof. At optimality, the gradient with respect to $\mathbf{u}$ of the above equation is zero. This implies that

$$2R\mathbf{u} + 2B^\top S B \mathbf{u} + 2B^\top S A \mathbf{x} = 0,$$

which implies

$$K = -(R + B^\top S B)^{-1} (B^\top S A).$$

□

Plugging this back into the Bellman equation, we obtain

$$\mathbf{x}^\top S \mathbf{x} = \mathbf{x}^\top Q \mathbf{x} + \mathbf{x}^\top K^\top R K \mathbf{x} + \mathbf{x}^\top (A + B K)^\top S (A + B K) \mathbf{x}.$$

Since this equality holds for all $\mathbf{x}$, we get

$$S = Q + K^\top R K + (A + B K)^\top S (A + B K).$$

Simplifying this expression we obtain

$$S = Q + A^\top S A - A^\top S B (R + B^\top S B)^{-1} B^\top S A.$$

This equation is called the **Discrete Time Algebraic Riccati Equation** (DARE) and is one of the most important equations in control theory.

5.3.1 Solving the DARE

In the one-dimensional case, although the right side of the equation is a rational function, we can multiply both sides by $R + B^\top SB$ to form a quadratic equation in S . Solving the DARE therefore corresponds to finding a non-negative root of a quadratic polynomial in the one-dimensional case. The reason we only consider the nonnegative root is that the value function is nonnegative by our choice of quadratic convex costs.

In higher dimension, the correct generalization is somewhat more difficult to see, but using Lemma 4.5, the equation can be reformulated as the solution to the following semidefinite program.

$$\begin{aligned} & \min_{X, S \in \mathcal{S}^{d_x \times d_x}} \langle X, I \rangle \\ & \text{subject to } \begin{bmatrix} X & B^\top SA \\ A^\top SB & R + B^\top SB \end{bmatrix} \succeq 0, \\ & \text{and } S = Q + A^\top SA - X. \end{aligned}$$

A popular alternative is to adapt the value iteration algorithm to this setting by iteratively applying the right side of the aforementioned equation as an operator. Concretely, starting from $S_0 \succeq 0$, we choose

$$S_{t+1} = Q + A^\top S_t A - A^\top S_t B (R + B^\top S_t B)^{-1} B^\top S_t A.$$

5.4 Robust Control

The solution to the LQR problem is a hallmark of optimal control theory and one of the most widely used techniques in control. However, the assumption that the disturbances are stochastic and i.i.d. is a strong one, and numerous attempts have been made over the years to generalize the noise model.

One of the most important and prominent generalizations is that of robust control, also known as H_∞ -control. The goal of H_∞ -control is to come up with a more robust solution, by considering the optimal solution vs. any possible perturbation sequence from a certain family. Formally, our aim is to solve the following:

$$\begin{aligned} & \min_{\mathbf{u}_{1:T}(\mathbf{x})} \max_{\mathbf{w}_{1:T}} \sum_{t=1}^T \mathbf{x}_t^\top Q \mathbf{x}_t + \mathbf{u}_t^\top R \mathbf{u}_t \\ & \text{s.t. } \mathbf{x}_{t+1} = A_t \mathbf{x}_t + B_t \mathbf{u}_t + \mathbf{w}_t, \end{aligned}$$

where we use the exact same notation as in previous sections.

Notice that this problem does not fall into the category of a Markov Decision Process, since $\mathbf{w}_t$'s are no longer stochastic. Therefore, the typical Bellman optimality criterion does not apply.

Instead, the appropriate techniques to address this problem come from the theory of dynamical games. The above is, in fact, a two-player zero sum game. The first player tries to minimize the costs, and the other attempts to maximize the costs, both players subject to the dynamics evolution constraint.

Using techniques from dynamic games, the following can be shown. References are provided in the bibliographic section.

Theorem 5.4. *For the H_∞ control problem with quadratic convex loss functions, the optimal solution is a linear policy of the state given by*

$$\mathbf{u}_t = K_t \mathbf{x}_t.$$

The proof is somewhat more involved than the LQR derivation and is omitted from this text.

5.5 Bibliographic Remarks

Optimal control is considered to be the hallmark of control theory, and numerous excellent introductory texts cover it in detail, e.g. [Ste94, Hes18].

The LQR problem and its solution were introduced in the seminal work of Kalman [Kal60], who also introduced the eponymous filtering procedure for linear estimation.

While seemingly a natural corollary of equivalent results for MDPs, the existence of value functions for linear control is made subtle by the continuous nature of the state space and stabilizability, or the lack thereof. A discussion of sufficient and necessary conditions for the existence of solutions to Riccati equations can be found in [Ber07, HSK96].

The development of H_∞ control was driven by an observation made in the seminal paper of Doyle [Doy78] that, unlike in the fully observed case, the solution to the optimal control problem for partially observed linear systems (which we study in later chapters) is not naturally robust to misspecification of system parameters. This suggests that robustness specifications constitute yet another axis of performance for control algorithms and may at times be at odds with the objective of optimal control. For in-depth treatment of H_∞ control, see [ZDG96b]. The text [BB08] presents a succinct derivation of robust controllers entirely in the state space, as the solution to a dynamic game between the controller and an adversary; see also [Tu18] for a recent exposition.

Although developed in the linear setting, adaptations of the LQR controller are also widely used to control nonlinear systems [Li05, LK13].

5.6 Exercises

- 5.1. Prove Theorem 5.2 for a time-varying linear dynamical system.
- 5.2. Derive the solution to the infinite-horizon LQR with zero mean Gaussian noise.
- 5.3. Extend the proof of Theorem 5.2 to positive semidefinite Q, R , rather than positive definite matrices.
- 5.4. Prove Theorem 5.4 as a dynamic two-player zero-sum game, where a unique saddle-point solution exists. Hint: you can formulate this problem as a min-max optimization problem within a finite horizon.
- 5.5. Prove that the function $f(\mathbf{x}) = \mathbf{x}^\top M \mathbf{x}$ is convex if and only if $M \succeq 0$ is positive semidefinite.

Part II

Basics of Online Control

Chapter 6

Regret in Control

In the first part of this book and in chapter 5 we studied linear dynamical systems with quadratic costs, namely the LQR problem. We saw that in this case the optimal control takes a very special form: it is a linear function of the state. The Markovian property of the Markov Decision Process framework for reinforcement learning suggests that the optimal policy is always a function of the current state. It is characterized by the Bellman optimality equation, and can be computed in complexity proportional to the number of states and actions using, for example, the value iteration algorithm, which we analyzed in a previous chapter.

However, this changes when we deviate from the MDP framework, which may be warranted for a variety of reasons:

1. The state is not fully observable.
2. The reward functions change online.
3. The transition mapping is not fixed and changes online.
4. The transitions or costs are subject to adversarial noise.

In such scenarios involving significant deviations from the MDP framework, classical notions of optimality may be ill-defined. Or if they do exist, the optimal policy may no longer be a mapping from immediate state to action. Rather, the learner may be required to use other signals available to her when choosing an action that is most beneficial in the long run. To this we add a significant consideration: the computation of optimal **state-based policies may be computationally intractable**.

This motivates the current chapter, which considers regret analysis as a computationally tractable alternative to optimal or robust control.

We start by considering regret in the decision-making framework of online convex optimization. We then consider several classes of control policies and the relationships between them in the context of regret analysis.

6.1 Online Convex Optimization

In this section, we give a brief introduction to the theory of Online Convex Optimization. The reader is referred to the bibliographic section for a comprehensive survey.

In the decision-making framework of Online Convex Optimization, a player iteratively chooses a point in a convex set $\mathcal{K} \subseteq \mathbb{R}^n$, and suffers a loss according to an adversarially chosen loss function $f_t : \mathcal{K} \mapsto \mathbb{R}$.

Let $\mathcal{A}$ be an algorithm for OCO, which maps a certain game history to a decision in the decision set:

$$\mathbf{x}_t^{\mathcal{A}} = \mathcal{A}(f_1, \dots, f_{t-1}) \in \mathcal{K}.$$

We define the regret of $\mathcal{A}$ after T iterations as:

$$\text{regret}_T(\mathcal{A}) = \sup_{\{f_1, \dots, f_T\} \subseteq \mathcal{F}} \left\{ \sum_{t=1}^T f_t(\mathbf{x}_t^{\mathcal{A}}) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}) \right\}. \quad (6.1)$$

If the algorithm is clear from the context, we henceforth omit the superscript and denote the algorithm's decision at time t simply as $\mathbf{x}_t$. Intuitively, an algorithm performs well if its regret is sublinear as a function of T (i.e. $\text{regret}_T(\mathcal{A}) = o(T)$), since this implies that, on average, the algorithm performs as well as the best fixed strategy in hindsight. We then describe a basic algorithm for online convex optimization.

6.1.1 Online Gradient Descent

Perhaps the simplest algorithm that applies to the most general setting of online convex optimization is Online Gradient Descent (OGD). The pseudocode is given in Algorithm 3.

In each iteration, the algorithm takes a step from the previous point in the direction of the gradient of the previous cost. This step may result in a point outside the underlying convex set. In such cases, the algorithm projects the point back to the convex set, i.e. finds its closest point in the convex set. Despite the fact that the next cost function may be completely different from the costs observed thus far, the regret attained by the algorithm is sublinear.

Algorithm 3 Online Gradient Descent

-
- 1: Input: convex set $\mathcal{K}$, T , $\mathbf{x}_1 \in \mathcal{K}$, step sizes $\{\eta_t\}$
 - 2: **for** $t = 1$ to T **do**
 - 3: Play $\mathbf{x}_t$ and observe cost $f_t(\mathbf{x}_t)$.
 - 4: Update and project:

$$\begin{aligned}\mathbf{y}_{t+1} &= \mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t) \\ \mathbf{x}_{t+1} &= \arg \min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\|^2 = \prod_{\mathcal{K}}[\mathbf{y}_{t+1}]\end{aligned}$$

- 5: **end for**

This is formalized in the following theorem. Here we define the constants D and G as follows:

- D = Euclidean diameter of the decision set, i.e.

$$\max_{\mathbf{x}, \mathbf{y} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|.$$

- G = an upper bound on the Lipschitz constant of the cost functions f_t , or equivalently, the subgradient norms of these functions. Mathematically,

$$\forall t, \quad \|\nabla f_t(\mathbf{x}_t)\| \leq G.$$

Theorem 6.1. *Online gradient descent with step sizes $\{\eta_t = \frac{D}{G\sqrt{t}}, t \in [T]\}$ guarantees the following for all $T \geq 1$:*

$$\text{regret}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x}^* \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}^*) \leq \frac{3}{2}GD\sqrt{T}.$$

Proof. Let $\mathbf{x}^* \in \arg \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x})$. Define $\nabla_t \stackrel{\text{def}}{=} \nabla f_t(\mathbf{x}_t)$. By convexity

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \leq \nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) \quad (6.2)$$

We first upper-bound $\nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*)$ using the update rule for $\mathbf{x}_{t+1}$ and the Pythagorean Theorem, we have the projections only decrease the distance to the set $\mathcal{K}$:

$$\|\mathbf{y}_{t+1} - \mathbf{x}^*\|^2 = \left\| \prod_{\mathcal{K}}(\mathbf{x}_t - \eta_t \nabla_t) - \mathbf{x}^* \right\|^2 \leq \|\mathbf{x}_t - \eta_t \nabla_t - \mathbf{x}^*\|^2. \quad (6.3)$$

Hence,

$$\begin{aligned} \|\mathbf{y}_{t+1} - \mathbf{x}^*\|^2 &\leq \|\mathbf{x}_t - \mathbf{x}^*\|^2 + \eta_t^2 \|\nabla_t\|^2 - 2\eta_t \nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) \\ 2\nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) &\leq \frac{\|\mathbf{x}_t - \mathbf{x}^*\|^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2}{\eta_t} + \eta_t G^2. \end{aligned} \quad (6.4)$$

Summing (6.2) and (6.4) from $t = 1$ to T , and setting $\eta_t = \frac{D}{G\sqrt{t}}$ (with $\frac{1}{\eta_0} \stackrel{\text{def}}{=} 0$):

$$\begin{aligned} 2 \left(\sum_{t=1}^T f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \right) &\leq 2 \sum_{t=1}^T \nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) \\ &\leq \sum_{t=1}^T \frac{\|\mathbf{x}_t - \mathbf{x}^*\|^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2}{\eta_t} + G^2 \sum_{t=1}^T \eta_t \\ &\leq \sum_{t=1}^T \|\mathbf{x}_t - \mathbf{x}^*\|^2 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) + G^2 \sum_{t=1}^T \eta_t \quad \frac{1}{\eta_0} \stackrel{\text{def}}{=} 0, \\ &\quad \|\mathbf{x}_{T+1} - \mathbf{x}^*\|^2 \geq 0 \\ &\leq D^2 \sum_{t=1}^T \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) + G^2 \sum_{t=1}^T \eta_t \\ &\leq D^2 \frac{1}{\eta_T} + G^2 \sum_{t=1}^T \eta_t \quad \text{telescoping series} \\ &\leq 3DG\sqrt{T}. \end{aligned}$$

The last inequality follows since $\eta_t = \frac{D}{G\sqrt{t}}$ and $\sum_{t=1}^T \frac{1}{\sqrt{t}} \leq 2\sqrt{T}$. $\square$

The Online Gradient Descent algorithm is straightforward to implement, and updates take linear time given the gradient. However, in general, the projection step may take significantly longer via convex optimization.

6.2 Regret for Control

The most important consideration to explore beyond classical models of control, such as optimal and robust control, is computational. Even in linear control with convex loss functions, the optimal policy can be complex to describe, see the exercises section.

A prominent approach to deal with computational difficulty is to consider a less stringent criterion for optimality. Instead of the optimal policy, we can

consider all policies in a certain *policy class*, and try to compete with the optimal policy within the class in terms of regret.

As alluded to earlier, the most natural classes of policies are computationally intractable. We thus resort to learning new classes that are computationally favorable. But to relate the performance of our methods, we require a language of comparing different policy classes. It is crucial to be able to relate the representation power of different policy classes. This motivates our main comparison metric as follows.

Definition 6.2. *We say that a class of policies Π_1 ε -approximates class Π_2 if the following is satisfied: for every $\pi_2 \in \Pi_2$, there exists $\pi_1 \in \Pi_1$ such that for all sequences of T disturbances and cost functions, it holds that*

$$\sum_{t=1}^T |c_t(\mathbf{x}_t^{\pi_1}, \mathbf{u}_t^{\pi_1}) - c_t(\mathbf{x}_t^{\pi_2}, \mathbf{u}_t^{\pi_2})| \leq T\varepsilon.$$

The significance of this definition is that if a simple policy class approximates another more complicated class, then it suffices to consider only the first. The usefulness of this definition comes from its implications for regret minimization, according to definition 1.3, as follows.

Lemma 6.3. *Let policy class Π_1 ε -approximate policy class Π_2 . Then for any control algorithm $\mathcal{A}$ we have*

$$\text{regret}_T(\mathcal{A}, \Pi_2) \leq \text{regret}_T(\mathcal{A}, \Pi_1) + \varepsilon T.$$

Proof. Recall that for policy class Π and algorithm $\mathcal{A}$,

$$\text{regret}_T(\mathcal{A}, \Pi) = \max_{\mathbf{w}_{1:T}: \|\mathbf{w}_t\| \leq 1} \left(\sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi \in \Pi} \sum_{t=1}^T c_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi) \right)$$

It follows from definition 6.2 that

$$\begin{aligned} \min_{\pi_2 \in \Pi_2} \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_2}, \mathbf{u}_t^{\pi_2}) &= \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_2^*}, \mathbf{u}_t^{\pi_2^*}) & \pi_2^* &= \arg \min_{\pi \in \Pi_2} \sum_t c_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi) \\ &\geq \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_1}, \mathbf{u}_t^{\pi_1}) - \varepsilon T & &\text{Definition 6.2} \\ &\geq \min_{\pi_1 \in \Pi_1} \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_1}, \mathbf{u}_t^{\pi_1}) - \varepsilon T \end{aligned}$$

Thus, for any fixed disturbance sequence $\mathbf{w}_{1:T}$,

$$\begin{aligned} & \sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi_2 \in \Pi_2} \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_2}, \mathbf{u}_t^{\pi_2}) \\ & \leq \sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \left(\min_{\pi_1 \in \Pi_1} \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_1}, \mathbf{u}_t^{\pi_1}) - \varepsilon T \right) \\ & = \left(\sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi_1 \in \Pi_1} \sum_{t=1}^T c_t(\mathbf{x}_t^{\pi_1}, \mathbf{u}_t^{\pi_1}) \right) + \varepsilon T \end{aligned}$$

Since this holds for any fixed disturbance sequence $\mathbf{w}_{1:T}$, it also holds for the maximum over all disturbance sequences $\mathbf{w}_{1:T}$ with $\|\mathbf{w}_t\| \leq 1$. $\square$

We now proceed to define and study approximation relationships between different policy classes.

6.3 Expressivity of Control Policy Classes

In this section, we formally relate the power of different complexity classes for linear time-invariant linear dynamical systems. The definitions and results can be extended to time-varying linear dynamical systems, and recall from Chapter 4 that time-varying linear dynamical systems were proposed by Lyapunov as a general methodology to study nonlinear dynamics.

For the rest of this section we assume full observation of the state and let

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t.$$

A sequence of stabilizing linear controllers for such a system is denoted by $\{K_t\}$. Schematically, the relationships we describe below are given in Figure 6.1 for fully observed systems. In later chapters of this text, we study policies for partially observed systems.

6.3.1 Linear Policies

The most basic and well-studied policy classes for control of linear dynamical systems are linear policies, i.e. those that choose a control signal as a (fixed) linear function of the state. The most important subcases of linear policies are those that are γ -stabilizing, meaning that the signal they create does not cause an unbounded state. This was formalized in Definition 2.4. Formally,

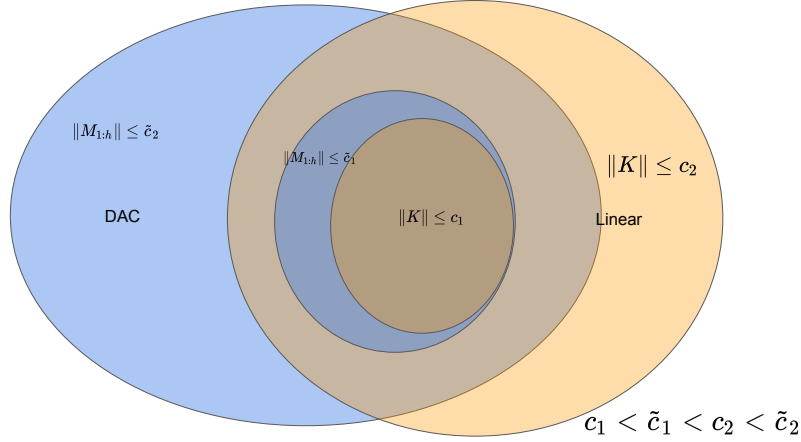


Figure 6.1: Schematic relationship between linear Disturbance-Action Control policies and linear state-action policies of increasing diameter for linear time invariant systems.

Definition 6.4 (Linear Policies). *A linear controller $K \in \mathbb{R}^{d_u \times d_x}$ is a linear operator that maps the state to the control. We denote by Π_γ^L the set of all linear controllers that are γ -stabilizing according to definition 2.4 and are bounded in Frobenius norm by $\|K\| \leq \gamma$.*

6.3.2 Disturbance Action Controllers

We next define the most important set of policies in online nonstochastic control, known as disturbance action controllers, or DAC. The main advantages of DAC, as opposed to linear controllers, are that the cost of control is convex in their parameterization and thus allows for provably efficient optimization. Moreover, this set of policies is stronger than linear policies in the sense that it ε -approximates them, as we show in the following.

Definition 6.5 (Disturbance Action Controller). *A (linear) disturbance action policy $\pi_{K_{1:T}, M_{1:h}}$ is parameterized by the matrices $M_{0:h} = [M_0, M_1, \dots, M_h]$ and stabilizing controllers $\{K_t\}$. It outputs control $\mathbf{u}_t^\pi$ at time t according to the rule*

$$\mathbf{u}_t^\pi = K_t \mathbf{x}_t + \sum_{i=1}^h M_i \mathbf{w}_{t-i} + M_0.$$

Without loss of generality, we take $M_0 = 0$, as we can add a fictitious unit coordinate to the perturbations $\{\mathbf{w}_t\}$, and modify the system accordingly. This will allow for a constant shift in the control.

Denote by $\Pi_{h,\gamma}^D$ the set of all disturbance action policies with $K_{1:T}$ being γ -stabilizing linear controllers for the system and

$$\sum_{i=1}^h \|M_i\| \leq \gamma.$$

The class of DACs always produces a stabilizing control signal whenever $K_{1:T}$ are sequentially stabilizing for the entire system $\{A_t, B_t\}$. This important fact can be seen in the following lemma.

Lemma 6.6. *Any DAC policy in $\Pi_{h,\gamma}^D$ is $2\gamma^2$ -stabilizing.*

Proof. For convenience, we define a fictitious noise term $\tilde{\mathbf{w}}_t = B \sum_{i=1}^h M_i \mathbf{w}_{t-i}$. First, due to the bound on $M_{1:h}$ and $\mathbf{w}_t$, we have that

$$\|\tilde{\mathbf{w}}_t\| = \left\| \sum_{i=1}^h M_i \mathbf{w}_{t-i} \right\| \leq \sum_{i=1}^h \|M_i\| \|\mathbf{w}_{t-i}\| \leq \gamma,$$

for $\|M\|$ being the spectral norm of M .

Next, since K_t 's are stabilizing, we can bound the historically distant terms as follows. For simplicity, assume that $K_t = K$ are all equal. Observe that the state at time t , following any DAC policy π , is given by

$$\begin{aligned} \mathbf{x}_{t+1} &= A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t \\ &= A\mathbf{x}_t + B \left(K\mathbf{x}_t + \sum_{i=1}^h M_i \mathbf{w}_{t-i} \right) + \mathbf{w}_t \\ &= (A + BK)\mathbf{x}_t + B \sum_{i=1}^h M_i \mathbf{w}_{t-i} + \mathbf{w}_t \\ &= \sum_{j=0}^t \left((A + BK)^j \left(B \sum_{i=0}^h M_i \mathbf{w}_{t-i-j} + \mathbf{w}_{t-j} \right) \right) \\ &= \sum_{j=0}^{t-1} \left((A + BK)^j (B\tilde{\mathbf{w}}_{t-j} + \mathbf{w}_{t-j}) \right). \end{aligned}$$

We can thus view the states $\mathbf{x}_t$ as generated by a γ -stabilizing policy over noises that are of magnitude 2γ , and hence bounded by $2\gamma^2$. $\square$

The above lemma is already a good indication about DAC: they generate controls that at least do not cause a blow-up. But how expressive are these policies in terms of reaching the optimal solution?

One of the main appeals of the DAC class is the following approximation lemma which shows that for time-invariant systems DAC are in fact a stronger class than linear controllers. Notice that the K_t terms for the DAC are only assumed to be stabilizing and not necessarily optimal for the actual sequence.

Lemma 6.7. *Consider a time-invariant linear dynamical system $\{A, B\}$. The class $\Pi_{h,\gamma}^D$ ε -approximates the class $\Pi_{\gamma'}^L$ for $h = \Omega(\gamma' \log(\frac{\gamma'}{\varepsilon}))$ and $\gamma = \gamma'^2$.*

Proof. Disturbance-action policies use a γ -stabilizing linear policy on which they superimpose linear functions of disturbances – fix such a policy, and let K_0 be this linear stabilizing component. The state sequence thus produced is identical to that obtained by running a DAC with $\mathbf{0}$ as the stabilizing controller on the system $(A + BK_0, B)$. Similarly, the state sequence produced by the execution of a linear policy K on (A, B) is identical to that of the linear policy $K - K_0$ on the system $(A + BK_0, B)$.

Lemma 6.8. *Let $\mathbf{x}_t^\pi(A, B), \mathbf{u}_t^\pi(A, B)$ be the state-action sequence produced by a policy π when executed on a linear dynamical system (A, B) . For any $\varepsilon > 0$, if two policies π_1, π_2 satisfy for all t that*

$$\|\mathbf{x}_t^{\pi_1}(A + BK_0, B) - \mathbf{x}_t^{\pi_2}(A + BK_0, B)\| \leq \varepsilon,$$

$$\|\mathbf{u}_t^{\pi_1}(A + BK_0, B) - \mathbf{u}_t^{\pi_2}(A + BK_0, B)\| \leq \varepsilon$$

when executed on some dynamical system $(A + BK_0, B)$, then when the corresponding policies π'_1 and π'_2 , that is, π_1 and π_2 with K_0 superimposed, are executed on the dynamical system (A, B) the iterates guarantee

$$\begin{aligned} & \max\{\|\mathbf{x}_t^{\pi'_1}(A, B) - \mathbf{x}_t^{\pi'_2}(A, B)\|, \\ & \|\mathbf{u}_t^{\pi'_1}(A, B) - \mathbf{u}_t^{\pi'_2}(A, B)\|\} \leq (1 + \|K_0\|)\varepsilon. \end{aligned}$$

With the aid of the above lemma, it will suffice to guarantee that the state-action sequence produced by the linear policy K and an appropriately chosen DAC with the stabilizing part set to $\mathbf{0}$ on an intrinsically γ -stable system (A, B) are similar enough. To prove this, we map any linear policy K to a disturbance-action policy that approximates it arbitrarily well for

some history parameter h . Specifically, for a given system A, B and linear policy K , let π_K be the policy given by

$$\forall i \in [0, h], \quad M_i = K(A + BK)^i.$$

Now consider any trajectory that results from applying the linear policy for the given system. Let $\mathbf{x}_{1:T}, \mathbf{u}_{1:T}$ be the states and controls of the linear policy K for a certain trajectory, as affected by a given noise signal. Let $\tilde{\mathbf{x}}_{1:T}, \tilde{\mathbf{u}}_{1:T}$ be the set of states and controls that would have been generated with the same noise sequence, but with the policy π_K . We first observe that control sequences produced by K and π_K are exponentially close.

$$\begin{aligned} \mathbf{u}_{t+1} &= K\mathbf{x}_{t+1} = K(A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t) \\ &= K(A + BK)\mathbf{x}_t + K\mathbf{w}_t \\ &= \sum_{i=0}^t K(A + BK)^i \mathbf{w}_{t-i} \end{aligned}$$

Now, we define $Z_t = \sum_{i=h+1}^t K(A + BK)^i \mathbf{w}_{t-i}$, and continue as shown.

$$\begin{aligned} \mathbf{u}_{t+1} &= \sum_{i=0}^h K(A + BK)^i \mathbf{w}_{t-i} + Z_t \\ &= \sum_{i=0}^h M_i \mathbf{w}_{t-i} + Z_t \\ &= \tilde{\mathbf{u}}_{t+1} + Z_t \end{aligned}$$

The magnitude of this residual term Z_t can be bounded, due to the existence of $\frac{\kappa}{\delta} \leq \gamma'$ from Exercise 4.5, as

$$\begin{aligned} |Z_t| &\leq \left\| \sum_{i=h+1}^t K(A + BK)^i \mathbf{w}_{t-i} \right\| \\ &\leq \sum_{i=h+1}^t \kappa(1 - \delta)^{i+1} \|K\| \|\mathbf{w}_{t-i}\| \\ &\leq \kappa^2 \sum_{i=h+1}^{\infty} (1 - \delta)^i \leq \kappa^2 \int_{i=h}^{\infty} e^{-\delta i} di \\ &= \kappa^2 \frac{1}{\delta} e^{-\delta h} \leq \varepsilon. \end{aligned}$$

We thus conclude that $\|\mathbf{u}_{t+1} - \tilde{\mathbf{u}}_{t+1}\| \leq \varepsilon$.

Similarly, for the corresponding state sequences, first note that

$$\mathbf{x}_{t+1} = \sum_{i=0}^t A^i (\mathbf{w}_{t-i} + B\mathbf{u}_{t-i}).$$

Therefore, since $\mathbf{x}_t$ and $\tilde{\mathbf{x}}_t$ share the same perturbation sequences as a part of their description, their difference can be bounded as a geometrically weighted sum of the difference in control inputs $\mathbf{u}_t$ and $\tilde{\mathbf{u}}_t$ that constitute them. Concretely, we have

$$\begin{aligned} \|\mathbf{x}_t - \tilde{\mathbf{x}}_t\| &= \left\| \sum_{i=1}^t A^{i-1} B (\mathbf{u}_{t-i} - \tilde{\mathbf{u}}_{t-i}) \right\| \\ &\leq \left\| \sum_{i=0}^t A^i B \right\| \max_{i \leq t} \|\mathbf{u}_i - \tilde{\mathbf{u}}_i\| \leq \frac{\kappa}{\delta} \varepsilon, \end{aligned}$$

by using the γ -stability of (A, B) . The approximation in terms of states and controls implies ε -approximation of policy class for any Lipschitz cost sequence, concluding the proof. $\square$

6.4 Bibliographic Remarks

Regret minimization, online learning, and online convex optimization have a rich history and have been studied for decades in the machine learning and game theory literature. The reader is referred to the comprehensive texts of [CBL06, Haz16] for a detailed treatment.

Linear state-feedback policies are a mainstay of control theory in both applications, due to their simplicity, and in theory, due to their optimality in deterministic and stochastic settings, as demonstrated in [Kal60]. The methods for computing such policies are often based on dynamic programming. Often motivated by the need to impose constraints beyond optimality, such as robustness, various other policy parameterizations were introduced. An early example is the Youla parameterization, sometimes called Q-parameterization [YJB76]. Later, the introduction of system-level synthesis (SLS) [DMW⁺17, ADLM19], in a similar vein, proved useful for distributed control, especially given its emphasis on state-space representations. For a comparison of these and additional ones such as input-output parameterizations [FZPK19], see [ZFP⁺20, ZZLL23].

The Disturbance Action Control parameterization, introduced in [ABH⁺19], is similar to system-level synthesis. Like SLS, it is a convex parameterization that is amenable to the tools of convex optimization. However, unlike SLS, it is a more compact representation that has logarithmically many parameters in its natural representation in terms of the desired approximation guarantee against linear policies. This makes it better suited for machine learning.

The study of regret in the context of control has roots in the work of [AYS11, AYBK14]. These works, as well as many that followed [DMM⁺18, MTR19, CHK⁺18, AHS19, SF20], study regret in a stochastic setting, where the perturbations are stochastic. In such settings, optimal control is well defined. The full power of regret analysis for control, involving both adversarial perturbations, adversarial loss functions, and computational considerations, was initiated in [ABH⁺19]. The notion of regret as applied to policy classes was implicit in the work of [EDKM04]. The regret metric when applied to policies is different from the counterfactual notion of policy regret defined by [ADT12].

6.5 Exercises

6.1. Let us say that the total number of time steps T is known in advance. Consider an implementation of online gradient descent where the step size is chosen as $\eta_t = D/G\sqrt{T}$, and therefore is uniform over time. Prove that the resultant regret is bounded by $GD\sqrt{T}$, which improves upon Theorem 6.1 by a constant factor. In fact, for ease of exposition, we will use this uniform step size scheme going forward.

6.2. In this exercise, we prove that the optimal solution to a linear control problem can be complex to describe.

1. Consider a linear dynamical system in one dimension and loss functions of the form $f_t(\mathbf{x}) = \max\{0, \mathbf{x} - c\}$, for $c_t \in \mathbb{R}$. Show that the optimal policy in hindsight can have as many domains as loss functions.
2. Consider a similar setting in higher dimensions and give a bound on the number of domains as a function of the number of loss functions and the dimension.

6.3. Consider the setting of time-varying linear dynamical systems, i.e.

$$\mathbf{x}_{t+1} = A_t \mathbf{x}_t + B_t \mathbf{u}_t + \mathbf{w}_t.$$

Prove that any DAC according to definition 6.5 is stable, i.e., prove the analogue of Lemma 6.6 for time-varying systems.

6.4. Prove Lemma 6.8.

Chapter 7

Online Nonstochastic Control

In this chapter, we explore recent advances in machine learning that have significantly reshaped the control problem. These developments mark a major departure from classical control theory and, in many cases, have led to methods that not only compete with, but often outperform traditional approaches. The key characteristics of these modern techniques are as follows:

1. **Online Learning Over Precomputed Policies** Unlike classical control, which relies on precomputing an optimal policy, online control adopts adaptive learning techniques to update policies dynamically. This aligns with the broader framework of adaptive control, where policies evolve in response to observed behavior of the system.
2. **Regret Minimization as a Performance Metric** While adaptive control focuses on learning a stabilizing policy, online control introduces a rigorous performance metric: regret, defined as the difference in cost between the learned policy and the best policy in hindsight. Unlike classical approaches, online nonstochastic control algorithms provide finite-time regret guarantees, ensuring competitive performance even under adversarial conditions.
3. **General Loss Functions and Adversarial Perturbations** Classical control typically assumes quadratic cost functions with known structures. In contrast, online nonstochastic control allows for general convex loss functions, which may be selected adversarially. This greater flexibility enables controllers to adapt to more diverse and complex real-world scenarios.

4. **Improper Learning via Convex Relaxation** A fundamental breakthrough in online nonstochastic control is the use of improper learning, where the algorithm competes against a given policy class while learning from a broader class of policies. Using convex relaxation techniques, these methods bypass computational hardness barriers that traditionally limit optimal control, and are accompanied by efficient algorithms with strong theoretical guarantees.

These innovations redefine how we approach control problems, shifting the focus from static policy optimization to adaptive, regret-minimizing strategies that operate effectively in dynamic and adversarial environments.

7.1 From Optimal and Robust to Online Control

Classical control theory has been mainly shaped by two paradigms: **optimal control** and **robust control**. These approaches have been instrumental in various engineering applications, but they rely on specific assumptions about disturbances and system dynamics.

- **Optimal control** assumes that disturbances follow a known stochastic model, often Gaussian. The objective is to design a control policy that minimizes the expected cumulative cost. Notable methods such as the Linear Quadratic Regulator (LQR) and stochastic dynamic programming fall into this category.
- **Robust control**, in contrast, takes a worst-case perspective. It ensures stability and performance even under the most adverse conditions, assuming perturbations belong to a predefined uncertainty set. A key example is H-infinity control.

Although these frameworks have been successful, real-world control tasks often involve uncertainties that do not fit neatly into either a probabilistic model or a worst-case adversarial setting. Consider the challenge of controlling a drone in an unpredictable environment:

- A stochastic model of wind disturbances may be unrealistic since actual wind patterns can exhibit long-term correlations or abrupt shifts.
- A worst-case robust approach may result in excessively conservative control, leading to inefficient and overly cautious maneuvers.

These limitations motivate the need for a more adaptive approach, one that can learn and adjust dynamically based on observed disturbances rather than relying on rigid assumptions. This leads us to **online nonstochastic control**, a fundamentally different paradigm where controllers adapt in real-time without assuming a fixed model for disturbances.

7.1.1 A Shift to Online Nonstochastic Control

Rather than relying on predefined stochastic or worst-case assumptions, online nonstochastic control introduces a new approach based on sequential decision-making and regret minimization. The core principles are the following.

1. **No Assumption on the Noise Model** – Disturbances (e.g., wind forces, external perturbations) are treated as an arbitrary sequence, potentially chosen adversarially. The controller must adapt dynamically rather than relying on a fixed distribution.
2. **Learning and Adaptation Over Time** – Instead of precomputing an optimal policy, online control algorithms iteratively refine their strategies using techniques from online convex optimization and sequential decision-making.
3. **Regret Minimization as the Performance Metric** – Unlike optimal control, which minimizes expected cost, or robust control, which prepares for the worst case, online control seeks to minimize regret. Regret quantifies how much worse the controller performs compared to the best policy in hindsight.

To build intuition, consider the problem of controlling an autonomous vehicle in an unknown city. If the vehicle had complete foresight of future traffic conditions, it could calculate the optimal route in advance. However, without such knowledge, the vehicle must continuously adjust its route based on real-time observations. Regret measures how far the vehicle's performance deviates from the best possible path that could have been chosen in hindsight.

This shift in perspective enables online controllers to handle a wide range of real-world disturbances, achieving adaptability, robustness, and computational efficiency without requiring strong assumptions about uncertainty.

In the following sections, we formally define the online nonstochastic control problem and introduce key algorithms that provide provable performance guarantees.

7.2 The Online Nonstochastic Control Problem

The shift from classical control to online nonstochastic control requires a fundamental rethinking of how we handle uncertainty. Unlike optimal control, which assumes disturbances follow a known probabilistic model, or robust control, which plans for the worst-case, online nonstochastic control operates in a more general and challenging setting.

In this framework, both disturbances and cost functions are chosen by an adversary, meaning that they may follow no statistically reliable pattern and could even be designed to hinder the controller's decisions at every step. As a result, the goal is no longer to compute a fixed optimal control policy, but rather to learn and adapt dynamically in response to an evolving sequence of perturbations and costs.

A fundamental difficulty in control is the computational tractability of finding the best policy. Classical control methods often rely on dynamic programming, which, in general, suffers from the curse of dimensionality. Even in simple settings, computing an optimal policy can be intractable due to the complexity of solving constrained optimization problems on trajectories. As pointed out by Rockafellar, finding an optimal control policy in arbitrary convex cost settings can be computationally prohibitive, particularly when state and control constraints are involved.

To address this, online nonstochastic control shifts the focus from computing an optimal policy to competing with a benchmark class of policies using regret minimization. Instead of optimizing a predefined objective function, the controller seeks to minimize its total loss relative to the best policy that could have been chosen in hindsight. This approach bypasses the need to solve an intractable dynamic optimization problem in advance while still achieving performance guarantees over time.

The fundamental challenge in online nonstochastic control can be summarized as follows:

- The controller does not have prior knowledge of the cost functions or disturbances.
- At each time step, the controller must select a control input based only on past observations.
- The performance of the controller is evaluated against a benchmark: the best fixed policy that could have been chosen in hindsight.

This formulation naturally connects to online learning and online convex optimization, where decisions must be made sequentially in uncertain environments. We now provide a formal definition of the problem.

Definition 7.1. *Consider a dynamical system evolving according to the equation: $\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t)$, where:*

- $\mathbf{x}_t \in \mathbb{R}^{d_{\mathbf{x}}}$ is the system state at time t ,
- $\mathbf{u}_t \in \mathbb{R}^{d_{\mathbf{u}}}$ is the control input at time t ,
- $\mathbf{w}_t \in \mathbb{R}^{d_{\mathbf{w}}}$ represents an external perturbation, which is chosen adversarially.

At each time step, the controller $\mathcal{A}$ selects $\mathbf{u}_t$ based on past observations, then incurs a cost defined by an arbitrary convex function $c_t : \mathbb{R}^{d_{\mathbf{x}}} \times \mathbb{R}^{d_{\mathbf{u}}} \rightarrow \mathbb{R}$.

The objective is to design a control strategy that minimizes regret, defined as the difference between the total cost incurred by the controller and the best fixed policy in hindsight:

$$\text{regret}_T(\mathcal{A}, \Pi) = \max_{\mathbf{w}_{1:T} : \|\mathbf{w}_t\| \leq 1} \left(\sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi \in \Pi} \sum_{t=1}^T c_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi) \right),$$

where Π is a benchmark class of policies, and $(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi)$ represent the state and control sequence obtained under the best policy π in hindsight.

Henceforth, if T, Π and $\mathcal{A}$ are clear from the context, we refer to regret without quantifiers. The goal of an online control algorithm is to achieve sublinear regret,

$$\text{regret}_T = o(T), \tag{7.1}$$

which ensures that, as $T \rightarrow \infty$, the controller performs competitively with the best fixed policy.

7.2.1 Improper Learning and Convex Relaxation

A key technique that enables online nonstochastic control algorithms to remain computationally efficient is improper learning through convex relaxation. Instead of directly competing with a predefined class of policies, these algorithms leverage a different policy class that allows for efficient optimization while still achieving strong performance guarantees.

This approach is crucial because enforcing exact constraints on the policy space can lead to computational hardness results, as previously discussed.

By relaxing the policy representation, for example, using convex approximations or surrogate loss functions, controllers can sidestep these intractabilities while still competing effectively with the best policy in hindsight.

In the next section, we introduce algorithmic solutions that utilize this principle to achieve low-regret online control under adversarial conditions.

7.3 The Gradient Perturbation Controller

After motivating the framework, defining regret, and exploring the relationships between different policy classes, we are ready to prove the main new algorithm in non-stochastic control.

The algorithm is described in figure (4), for time-varying linear dynamical systems. It is a specialization of the template introduced in Algorithm 1 from Section 1.6.2. The main idea is to learn the parameterization of a DAC policy using known online convex optimization algorithms, namely online gradient descent. We henceforth prove that this parameterization is convex, allowing us to prove the main regret bound.

The notation we use to describe the algorithm is as follows. We denote by $\mathbf{x}_t(M_{1:h})$ the hypothetical state reached by playing a DAC policy $M_{1:h}$ from the beginning of time. Note that this may not be equal to the true state $\mathbf{x}_t$ of the dynamical system at time t . The counterfactual $\mathbf{u}_t(M_{1:h})$ is defined similarly.

Algorithm 4 Gradient Perturbation Controller(GPC)

- 1: Input: h, η , initialization $M_{1:h}^1 \in \mathcal{K}$.
 - 2: Observe the linear system (A_t, B_t) , compute a stabilizing linear controller K_t as per Theorem 4.4.
 - 3: **for** $t = 1 \dots T$ **do**
 - 4: Use control $\mathbf{u}_t = K_t \mathbf{x}_t + \sum_{i=1}^h M_i^t \mathbf{w}_{t-i}$.
 - 5: Observe state $\mathbf{x}_{t+1}$; compute noise $\mathbf{w}_t = \mathbf{x}_{t+1} - A_t \mathbf{x}_t - B_t \mathbf{u}_t$.
 - 6: Record an instantaneous cost $c_t(\mathbf{x}_t, \mathbf{u}_t)$.
 - 7: Construct loss $\ell_t(M_{1:h}) = c_t(\mathbf{x}_t(M_{1:h}), \mathbf{u}_t(M_{1:h}))$.
 Update $M_{1:h}^{t+1} \leftarrow \prod_{\mathcal{K}} [M_{1:h}^t - \eta \nabla \ell_t(M_{1:h}^t)]$.
 - 8: **end for**
-

The GPC algorithm comes with a very strong performance guarantee: it guarantees vanishing regret vs. the best Disturbance Action Control policy belonging to our reference class in hindsight. According to the relationship

between DAC and other policies, this implies vanishing regret versus linear controllers as well. The formal statement is given as follows.

Theorem 7.2. *Assuming that for all $t \in [T]$*

- a The costs c_t are convex, bounded, and L -Lipschitz.*
- b The matrices $\{A_t, B_t\}$ are γ -stabilizable and have a bounded ℓ_2 norm.*

Then the GPC algorithm (4) ensures that

$$\max_{\mathbf{w}_{1:T}: \|\mathbf{w}_t\| \leq 1} \left(\sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi \in \Pi^D} \sum_{t=1}^T c_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi) \right) \leq \mathcal{O}(LGD\gamma^2 h \sqrt{T}),$$

where the $\mathcal{O}$ notation hides constants that only depend on the norm of A_t, B_t . Furthermore, the time complexity of each loop of the algorithm is linear in the number of system parameters and logarithmic in T .

Note: we henceforth assume that the linear dynamical systems (A_t, B_t) are a linear time-invariant system (A, B) which is γ -stable, and there is no need for a stabilizing controller $K_t = 0$. The extension to time-varying stabilizable systems is left as exercise 7.5.

Proof. According to Theorem 6.1, the online gradient descent algorithm for convex loss functions ensures that the regret is bounded by $2GD\sqrt{T}$. However, to directly apply the OGD algorithm to our setting, we need to ensure that the following conditions hold true:

- The loss function should be a convex function in the variables $M_{1:h}$.
- We must ensure that optimizing the loss function ℓ_t as a function of parameters $M_{1:h}$ is similar to optimizing the cost c_t as a function of the state and control.

These facts are formalize in the following two lemmas, which we state here and prove at the end of this section.

Lemma 7.3. *The loss functions ℓ_t are convex in the variables $M_{1:h}$.*

Lemma 7.4. *Let G denote a upper bound on the gradients of the L -Lipschitz losses c_t , and D denote the diameter of the matrices $M_{1:h}$. Then there exists a constant C that depends only on $\|A\|, \|B\|$, such that for all t ,*

$$|l_t(M_{1:h}^t) - c_t(\mathbf{x}_t, \mathbf{u}_t)| \leq \frac{CL\gamma^2 h D}{\sqrt{T}}.$$

The regret bound of GPC can be derived from these lemmas as follows. Note that G is an upper bound on the gradients of ℓ_t as a function of the parameters $M_{1:h}$, while L is the Lipschitz constant of c_t as a function of the state and control.

$$\begin{aligned}
& \sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi \in \Pi^D} \sum_{t=1}^T c_t(\mathbf{x}_t^\pi, \mathbf{u}_t^\pi) \\
& \leq \sum_{t=1}^T \ell_t(M_{1:h}^t) - \min_{\pi \in \Pi^{DAC}} \ell_t(M_{1:h}^\pi) + T \times \frac{L\gamma^2 ChD}{\sqrt{T}} \quad \text{Lemma 7.4} \\
& \leq 2GD\sqrt{T} + \sqrt{T}L\gamma^2 ChD \quad \text{Theorem 6.1} \\
& \leq O(GDL\gamma^2 h\sqrt{T}).
\end{aligned}$$

□

7.3.1 Convexity of the Losses

To conclude the analysis, we first show that the loss functions are convex with respect to the variables $M_{1:h}$. This follows since the states and the controls are linear transformations of the variables.

Proof of Lemma 7.3. For simplicity, assume that the initial state $\mathbf{x}_0$ is $\mathbf{0}$. Using $\mathbf{x}_t$ to denote $\mathbf{x}_t(M_{1:h})$ and $\mathbf{u}_t$ to denote $\mathbf{u}_t(M_{1:h})$, the loss function ℓ_t is given by

$$\ell_t(M_{1:h}) = c_t(\mathbf{x}_t, \mathbf{u}_t).$$

Since the cost c_t is a convex function with respect to its arguments, we simply need to show that $\mathbf{x}_t$ and $\mathbf{u}_t$ depend linearly on $M_{1:h}$. The state is given by

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t = A\mathbf{x}_t + B \left(\sum_{i=1}^h M_i \mathbf{w}_{t-i} \right) + \mathbf{w}_t$$

By induction, we can further simplify

$$\mathbf{x}_{t+1} = \sum_{i=0}^t A^i \left(B \sum_{j=1}^h M_j \mathbf{w}_{t-i-j} + \mathbf{w}_{t-i} \right), \quad (7.2)$$

which is a linear function of the variables.

Similarly, the control $\mathbf{u}_t$ is given by

$$\mathbf{u}_t = \sum_{i=1}^h M_i \mathbf{w}_{t-i}.$$

Thus, we have shown that $\mathbf{x}_t$ and $\mathbf{u}_t$ are linear transformations of $M_{1:h}$. A composition of convex and linear functions is convex (see Exercise 7.1), which concludes our lemma. $\square$

7.3.2 Loss Functions with Memory

The actual loss c_t at time t is not calculated on $\mathbf{x}_t(M_{1:h})$, but rather on the true state $\mathbf{x}_t$, which in turn depends on different parameters $M_{1:h}^i$ for various historical times $i < t$. However, $c_t(\mathbf{x}_t, \mathbf{u}_t)$ is well approximated by $l_t(M_{1:h}^t)$, as shown next.

Proof of Lemma 7.4. We first note that by the choice of step size by the Online Gradient Descent algorithm 3, as per Theorem 6.1, we have that $\eta = \frac{D}{G\sqrt{T}}$. Thus,

$$\|M_{1:h}^t - M_{1:h}^{t-i}\| \leq \sum_{s=t-i+1}^t \|M_{1:h}^s - M_{1:h}^{s-1}\| \leq i\eta G = \frac{iD}{\sqrt{T}}.$$

We next use this fact to establish that $\mathbf{x}_t$ and $\mathbf{x}_t(M_{1:h}^t)$ are close. Similar to the expansion of $\mathbf{x}_t(M_{1:h})$ in Equation 7.2, $\mathbf{x}_t$ can be written as

$$\mathbf{x}_{t+1} = \sum_{i=0}^t A^i \left(B \sum_{j=1}^h M_j^{t-i} \mathbf{w}_{t-i-j} + \mathbf{w}_{t-i} \right).$$

Recall that the stability of the time-invariant linear dynamical system $\{A, B\}$ implies that for any i , $\|A^i\| \leq \kappa(1-\delta)^i$, with $\frac{\kappa}{\delta} \leq \gamma$. Therefore, there exists a constant $C \geq 0$ such that

$$\begin{aligned} \|\mathbf{x}_t - \mathbf{x}_t(M_{1:h}^t)\| &= \left\| \sum_{i=0}^t A^i B \sum_{j=1}^h M_j^{t-i} \mathbf{w}_{t-i-j} - \sum_{i=0}^t A^i B \sum_{j=1}^h M_j^t \mathbf{w}_{t-i-j} \right\| \\ &\leq \sum_{i=0}^t \|A^i\| \|B\| \sum_{j=1}^h \|M_j^{t-i} - M_j^t\| \|\mathbf{w}_{t-i-j}\| \\ &\leq C \cdot \kappa \sum_{i=0}^t (1-\delta)^i \cdot i \frac{hD}{\sqrt{T}} \\ &= \frac{Ch\kappa D}{\sqrt{T}} \sum_{i=1}^{\infty} i(1-\delta)^{i-1} \\ &\leq \frac{C\gamma^2 hD}{\sqrt{T}}, \end{aligned}$$

where on the second last line we use $\sum_{i=1}^{\infty} i(1-\delta)^i = \frac{1}{\delta^2}$.

The control at time t depends only on the parameters at the current iteration and not on historical parameters, hence $\mathbf{u}_t = \mathbf{u}_t(M_{1:h}^t)$. By definition, $l_t(M_{1:h}^t) = c_t(\mathbf{x}_t(M_{1:h}^t), \mathbf{u}_t(M_{1:h}^t))$. Thus, we have

$$\begin{aligned} l_t(M_{1:h}^t) - c_t(\mathbf{x}_t, \mathbf{u}_t) &= c_t(\mathbf{x}_t(M_{1:h}^t), \mathbf{u}_t(M_{1:h}^t)) - c_t(\mathbf{x}_t, \mathbf{u}_t) \\ &\leq L \cdot \|\mathbf{x}_t(M_{1:h}^t) - \mathbf{x}_t\| \leq \frac{L\gamma^2 ChD}{\sqrt{T}}. \end{aligned}$$

□

7.4 Bibliographic Remarks

Online nonstochastic control is an extension of the framework of online convex optimization [Haz16], as it is applied to policies, and thus to loss functions with memory [AHM13].

The bulk of the material in this chapter is from [ABH⁺19]. The work of [ABH⁺19] defined the general online nonstochastic setting in which the disturbances and the cost functions are adversarially chosen, and where the cost functions are arbitrary convex functions.

The follow-up work by [AHS19] achieves a logarithmic pseudo-regret for strongly convex, adversarially selected losses, and well-conditioned stochastic noise. Logarithmic regret algorithms for the full setting of online nonstochastic control with strongly convex costs were obtained in [Sim20, FS20, CCK20].

Online nonstochastic control with known dynamics was further extended in several directions. The task of control with bandit feedback for the costs was considered in [GHH20]. More refined regret metrics applicable to changing dynamics were studied in [GHM20, MGS21, ZWZ22, Sin22]. More extensions suitable for changing environments were studied in [BW22]. Online nonstochastic control of population dynamics was studied in [GHL⁺24].

[LDL21] studies algorithms that balance regret minimization while adhering to constraints. An application of nonstochastic control in iterative planning in the presence of perturbations is described in [AHMS21]. Various open-source software packages, such as [GHM20], provide benchmarked implementations of many such algorithms.

An alternate performance metric for online control, which is also nonstochastic, is the competitive ratio. This objective tracks the cost of the online controller and compares it to the best policy in hindsight. Control methods that minimize the competitive ratio were initiated in [GW19]. This approach was extended in a series of papers [GLSW19, SLC⁺20, GH21]. Thus far, the success of this approach was limited to fully observed linear dynamical systems and further to quadratic costs when achieving the optimal competitive ratio. [GASH22] show that regret minimizers in the non-stochastic setting automatically achieve a near-optimal competitive ratio.

A precursor to online nonstochastic control is a line of work, beginning with [AYBK14], focused on the control of stochastic systems under sequentially revealed adversarial costs functions. See [CK20, CCK22] for further developments.

7.5 Exercises

7.1. Let $l : \mathbb{R}^d \rightarrow \mathbb{R}$ be a convex function. Prove that for any matrix $A \in \mathbb{R}^{d \times n}$, $l(Ax)$ is convex for $x \in \mathbb{R}^n$.

7.2.

1. Let $l : \mathbb{R}^d \rightarrow \mathbb{R}$ be a β -smooth function and let $A \in \mathbb{R}^{d \times n}$ be a matrix. Show that $l(Ax)$ is necessarily a smooth function for $x \in \mathbb{R}^n$.
2. Let $l : \mathbb{R}^d \rightarrow \mathbb{R}$ be a μ -strongly convex function and let $A \in \mathbb{R}^{d \times n}$ be a matrix. Show that $l(Ax)$ is not necessarily strongly-convex for $x \in \mathbb{R}^n$.

7.3. Prove that the loss functions ℓ_t are convex in $M_{1:h}$, as per Lemma 7.3, even if the initial state is non-zero.

7.4. Consider control of a general, non-linear dynamical system. Give a version of the GPC algorithm 4 that can be applied to such a setting (without proving any performance guarantee).

7.5. In this exercise, we prove and extend Theorem 7.2 for linear time-varying systems that are stabilizable, rather than stable.

1. Write down and prove the theorem for time-varying systems $\{A_t, B_t\}$ that are all γ -stable.
2. Consider a sequence of γ -stabilizable time-varying systems $\{A_t, B_t\}$. Show how to reduce this case to the previous by considering a sequence of linear time-varying γ -stable systems $\tilde{A}_t, B_t$, where $\tilde{A}_t = A_t + B_t K_t$.

Chapter 8

Online Nonstochastic System Identification

In the previous chapters, we focused on control algorithms that assume knowledge of the system dynamics. However, in many real-world applications, system parameters are unknown or change over time. This necessitates an approach in which the controller must simultaneously learn the system model while making control decisions. This problem, known as system identification, is fundamental to adaptive control, reinforcement learning, and modern data-driven methods for control.

8.1 The Role of System Identification in Online Control

System identification plays a crucial role in control design, as it allows controllers to estimate unknown system parameters in real time. Unlike classical identification methods that rely on extensive offline data collection, an online approach must identify the system while actively controlling it. This presents several unique challenges:

- Feedback loops and nonstationarity: Unlike supervised learning settings, where data is collected passively, control decisions influence future observations, making the estimation process highly dynamic.
- Adversarial or arbitrary disturbances: The disturbances affecting the system may not follow a probabilistic model but can be chosen adversarially, leading to challenges in maintaining robustness.

- Balancing exploration and performance: The controller must explore sufficiently to estimate system parameters accurately while ensuring stable performance, a trade-off that directly impacts regret minimization.

8.2 A Nonstochastic Approach to System Identification

Traditional system identification techniques often rely on stochastic assumptions, where noise is assumed to be drawn from a known distribution, and system parameters are estimated using statistical inference techniques. However, in nonstochastic system identification, no assumptions are made about the noise distribution or system perturbations. Instead, identification strategies are designed to be robust to worst-case disturbances and operate under adversarial uncertainty.

This shift in perspective has several implications:

- Performance guarantees without probabilistic assumptions: Unlike stochastic estimation methods, which provide guarantees under specific noise models, nonstochastic approaches aim for regret bounds that hold in arbitrary environments.
- Regret as a measure of learning efficiency: The effectiveness of an identification strategy is measured not in terms of statistical efficiency but in its ability to achieve low regret, ensuring that the system is identified quickly enough to enable near-optimal control.
- Computational tractability in identification: Classical system identification methods often involve solving non-convex optimization problems, which makes them computationally demanding. Nonstochastic approaches frequently leverage convex relaxation techniques to sidestep intractability.

The problem of online system identification is closely related to online convex optimization and sequential decision making. Many regret minimization techniques developed for online learning can be adapted to system identification settings. However, a key distinction is that, in control settings, the learned system model directly affects future decisions, creating a coupling between estimation and control that does not typically exist in standard online learning formulations.

This connection between system identification and online learning leads to several fundamental questions:

- How should a controller optimally trade off exploration (learning system parameters) and exploitation (minimizing control cost)?
- What are the best algorithmic strategies for estimating system parameters while controlling in adversarial settings?
- What regret guarantees can be achieved in online nonstochastic system identification, and how do they compare to traditional learning-based control approaches?

8.3 Nonstochastic System Identification

The problem we study in this section is online nonstochastic system identification of a linear time-invariant dynamical system.

As a first step, we consider the question of whether the system matrices A, B can be recovered from observations over the state, without regard to the actual cost of control.

The method below to achieve this learns the system matrices by using a random control. Specifically, at each step the control $\mathbf{u}_t$ will be randomly selected from a Rademacher distribution, according to Algorithm 5.

Algorithm 5 System identification via method of moments.

Input: controllability index k , oversampling parameter T_0 .

for $t = 0, \dots, T_0(k+1)$ **do**

 Execute the control $\mathbf{u}_t = \eta_t$ with $\eta_t \sim_{i.i.d.} \{\pm 1\}^d$.

 Record the observed state $\mathbf{x}_t$.

end for

Set $\widehat{A}^j B = \frac{1}{T_0} \sum_{t=0}^{T_0-1} \mathbf{x}_{t(k+1)+j+1} \eta_{t(k+1)}^\top$, for all $j \in \{0, \dots, k\}$.

Define $\widehat{C}_0 = (\widehat{B}, \dots, \widehat{A}^{k-1} B)$, $\widehat{C}_1 = (\widehat{A} B, \dots, \widehat{A}^k B)$, and return $\widehat{A}, \widehat{B}$ as

$$\widehat{B} = \widehat{B}, \quad \widehat{A} = \arg \min_A \|\widehat{C}_1 - A \widehat{C}_0\|_F.$$

The main guarantee from this simple system identification procedure is given as follows.

Theorem 8.1. *Assuming that*

- a the system (A, B) is γ -stable,
- b the system is (k, κ) -strongly controllable,
- c the perturbations $\mathbf{w}_t$ do not depend on $\mathbf{u}_t$,

and for $T_0 = \Omega\left(\frac{d^2 k^2 \kappa^2 \gamma^6}{\varepsilon^2} \log \frac{1}{\Delta}\right)$, the following holds. Algorithm 5 outputs a pair $(\widehat{A}, \widehat{B})$ that satisfies, with probability $1 - \Delta$,

$$\|\widehat{A} - A\| \leq \varepsilon, \|\widehat{B} - B\| \leq \varepsilon.$$

The proof of this theorem hinges upon the key observation that

$$\mathbf{E}[\mathbf{x}_{t+k} \mathbf{u}_t^\top] = \mathbf{E}\left[\sum_{i=0:k} A^i (B \mathbf{u}_{t+k-i} + \mathbf{w}_t) \mathbf{u}_t^\top\right] = A^k B,$$

and the crucial fact that the controls are i.i.d. Thus, we can create an unbiased estimator for this matrix, and using many samples, we obtain an increasingly accurate estimate.

Proof. First, we establish that $A^j B$ is close to $\widehat{A^j B}$ for all $j \in [k]$.

Lemma 8.2. *As long as $T_0 \geq \widetilde{\Omega}\left(\frac{d^2 \gamma^2}{\varepsilon^2} \log \frac{1}{\Delta}\right)$, with probability $1 - \Delta$, we have for all $j \in [k]$ that*

$$\|\widehat{A^j B} - A^j B\| \leq \varepsilon.$$

Proof. First note that for any t , since the system is γ stable and the controls have norm $\sqrt{d}$, the state $\mathbf{x}_t$ is bounded as

$$\|\mathbf{x}_t\| = \left\| \sum_{i=1}^t A^{i-1} (B \eta_{t-i} + \mathbf{w}_{t-i}) \right\| \leq 2\sqrt{d}\gamma.$$

Therefore, defining $N_{j,t} = \mathbf{x}_{t(k+1)+j+1} \eta_{t(k+1)}^\top$ for any $j < k$ and $t < T_0$, we have that $\|N_{j,t}\| \leq 2d\gamma$. Note that $\{N_{j,t}\}_{t \leq T_0}$ is not a sequence of independent random variables, since the same instance of η_t might occur in multiple of these terms. To remedy this, we claim that the sequence $\tilde{N}_{j,t} = N_{j,t} - A^j B$ forms a martingale difference sequence with respect to the filtration Σ_t , defined as $\Sigma_t = \{\eta_i : i < (k+1)(t+1)\}$. Since $N_{j,t}$ only involves the first $t(k+1) + k$ random control inputs, $N_{j,t}$ is Σ_t -measurable. Now, for any t ,

since η_t is zero-mean and chosen independently of $\{w_j\}, \{\eta_j\}_{j \neq t}$ at each time step, we have

$$\begin{aligned} \mathbb{E}[N_{j,t}|\Sigma_{t-1}] &= \mathbb{E}\left[\sum_{i>1} A^{i-1} (\mathbf{w}_{t(k+1)+j+1-i} + B\eta_{t(k+1)+j+1-i}) \eta_{t(k+1)}^\top \middle| \Sigma_{t-1}\right] \\ &= \mathbb{E}\left[\sum_{i>1} A^{i-1} B\eta_{t(k+1)+j+1-i} \eta_{t(k+1)}^\top \mathbb{I}_{i=j+1} \middle| \Sigma_{t-1}\right] = A^j B, \end{aligned}$$

where the last equality follows from the fact that $\mathbb{E}[\eta_t \eta_t^\top] = I$.

Theorem 8.3 (Matrix Azuma). *Consider a sequence of random symmetric matrices $\{X_k\} \in \mathbb{R}^{d \times d}$ adapted to a filtration sequence $\{\Sigma_k\}$ such that*

$$\mathbb{E}[X_k|\Sigma_{k-1}] = 0 \quad \text{and} \quad X_k^2 \preceq \sigma^2 I \text{ holds almost surely,}$$

then, for all $\varepsilon > 0$, we have

$$P\left(\lambda_{\max}\left(\frac{1}{K} \sum_{k=1}^K X_k\right) \geq \varepsilon\right) \leq de^{-K\varepsilon^2/8\sigma^2}.$$

Applying the Matrix Azuma inequality (see the bibliography for a reference) on the symmetric dilation, i.e.

$$\begin{bmatrix} 0 & \tilde{N}_{j,t}^\top \\ \tilde{N}_{j,t} & 0 \end{bmatrix},$$

we have for all $j \in [k]$ with probability $1 - \Delta$ that

$$\|\widehat{A^j B} - A^j B\| \leq 8d\gamma \sqrt{\frac{1}{T_0} \log \frac{n+d}{\Delta}}.$$

A union bound over $j \in [k]$ concludes the claim. $\square$

At this point, define $C_0 = (B, AB, \dots, A^{k-1}B)$ and $C_1 = (AB, A^2B, \dots, A^k B)$ to observe that with probability $1 - \Delta$, we have that

$$\|\widehat{C_0} - C_0\|, \|\widehat{C_1} - C_1\| \leq \sqrt{k}\varepsilon.$$

By this display, we already have $\|B - \widehat{B}\| \leq \varepsilon$. We will make use of this observation to establish the proximity of $\widehat{A}$ and A , using the following lemma, the proof of which is left as Exercise 8.3.

Lemma 8.4. *Let $X, X + \Delta X$ be the solutions to the linear system of equations $XA = B$ and $(X + \Delta X)(A + \Delta A) = B + \Delta B$. Then, as long as $\|\Delta A\| < \sigma_{\min}(A)$, we have*

$$\|\Delta X\| \leq \frac{\|\Delta B\| + \|\Delta A\|\|X\|}{\sigma_{\min}(A) - \|\Delta A\|}$$

Note that $\widehat{A}$ does not necessarily satisfy $\widehat{A}\widehat{C}_0 = \widehat{C}_1$. We now use the above lemma on the normal equations, that is, on $AC_0C_0^\top = C_1C_0^\top$ and $\widehat{A}\widehat{C}_0\widehat{C}_0^\top = \widehat{C}_1\widehat{C}_0^\top$. First, observe that for any $\varepsilon < 1$

$$\begin{aligned} \|C_0C_0^\top - \widehat{C}_0\widehat{C}_0^\top\| &= \|C_0C_0^\top - C_0\widehat{C}_0^\top + C_0\widehat{C}_0^\top - \widehat{C}_0\widehat{C}_0^\top\| \\ &\leq \|C_0\|\|C_0 - \widehat{C}_0\| + \|\widehat{C}_0\|\|C_0 - \widehat{C}_0\| \\ &\leq 2\|C_0\|\sqrt{k}\varepsilon + k\varepsilon^2 \leq 3\gamma k\varepsilon, \end{aligned}$$

where the last line uses that $\|C_0\| \leq \gamma$ following the stability of (A, B) . A similar display implies the same upper bound on $\|C_1C_1^\top - \widehat{C}_1\widehat{C}_1^\top\|$. Therefore, using $\|A\| \leq \gamma$ and the perturbation result for linear systems, we have

$$\|A - \widehat{A}\| \leq \frac{12\gamma^2k}{\sigma_{\min}(C_0C_0^\top)}\varepsilon,$$

as long as $6k\gamma\varepsilon < \sigma_{\min}(C_0C_0^\top)$. Finally, note that (k, κ) -strong controllability implies $\sigma_{\min}(C_0C_0^\top) \geq 1/\kappa$. $\square$

8.4 From Identification to Nonstochastic Control

Having recovered approximate estimates $(\widehat{A}, \widehat{B})$ of the system matrices, a natural question arises: how well does the Gradient Perturbation Algorithm 4 perform on the original system (A, B) when provided with only approximate estimates of the same? We briefly state a result to this extent, whose proof we leave to the explorations of an interested reader.

Lemma 8.5. *Given estimates of the system matrices $(\widehat{A}, \widehat{B})$ such that*

$$\|\widehat{A} - A\|, \|\widehat{B} - B\| \leq \varepsilon,$$

the GPC algorithm when run on the original system (A, B) guarantees a regret bound of

$$\max_{\mathbf{w}_{1:T}: \|\mathbf{w}_t\| \leq 1} \left(\sum_{t=1}^T c_t(\mathbf{x}_t, \mathbf{u}_t) - \min_{\pi \in \Pi^{DAC}} \sum_{t=1}^T c_t(\hat{\mathbf{x}}_t, \pi(\hat{\mathbf{x}}_t)) \right) \leq \mathcal{O}(\sqrt{T} + \varepsilon T).$$

This result translates to a regret bound of $\tilde{O}(T^{2/3})$ for nonstochastic control of unknown, yet stable, linear systems, by first running the nonstochastic system identification procedure, and subsequently using the estimates thus obtained to run GPC. Details of how to derive this regret bound are left as Exercise 8.1.

8.5 Summary and Key Takeaways

This chapter explored online nonstochastic system identification, where a controller must simultaneously learn an unknown system model while making control decisions in an adversarial setting. Unlike classical stochastic system identification, which assumes structured noise and statistical convergence, the nonstochastic approach makes no such assumptions and instead relies on regret minimization techniques to ensure efficient learning.

The key insights from this chapter are as follows.

- **The Exploration-Exploitation Tradeoff** A fundamental challenge in online system identification is balancing exploration, which enables learning the system dynamics, and exploitation, which minimizes control cost. Too much exploration leads to high immediate costs, while too much exploitation can result in poor long-term learning and sub-optimal control performance. The algorithms in this chapter address this by incorporating controlled perturbations in the control inputs, ensuring sufficient exploration while keeping regret low. However, the trade-off results in a regret of $O(T^{2/3})$, rather than $O(\sqrt{T})$ we had in the last chapter.

More sophisticated techniques for balancing exploration and exploitation can result in better regret bounds, some of which are surveyed in the bibliographic section.

- **Regret Minimization as a Learning Metric** Unlike traditional identification methods that aim at asymptotic convergence, online nonstochastic identification is evaluated in terms of regret, measuring the performance relative to the best fixed model in hindsight.

8.5.1 Looking Ahead

This part of the book considered the online nonstochastic control problem, the regret metric, and online learning techniques applied to control. We have

thus far considered the case where the state is observable to the controller, an assumption which does not always hold.

In the next parts of the book, we consider the setting of partial observability, where the system state is unknown, but an observed signal is available. We start with the problem of learning and move onto online control.

8.6 Bibliographic Remarks

Controlling linear dynamical systems given unknown or uncertain system parameters has long been studied in adaptive control; see the text of [SBB90] for example. However, the results from adaptive control typically concern themselves with stability and, even when discussing implications for optimality, present asymptotic bounds.

In contrast, much of the work at the intersection of machine learning and control is focussed on precise non-asymptotic bounds on sample complexity or regret. [Fie97] which studies the discounted LQR problem under **stochastic noise** is the earliest example that provides a precise sample complexity bound. Similarly, the early work of [AYS11] uses an optimism-based approach to achieve optimal regret in terms of dependence on horizon. This regret bound was later improved for sparse systems in [IJR12]. [DMM⁺18, DMM⁺20] established regret and sample complexity results for the general stochastic LQR formulation with a polynomial dependence on natural system parameters. [FGKM18] studies the efficacy of first-order optimization-based approaches in synthesizing controllers for the LQR problem. The work of [MTR19] establishes that simple explore-then-commit strategies achieve the optimal dependence of regret on horizon, with [SF20] proving that such approaches are optimal in terms of their dependence on dimension of the system, as well. [CKM19] also gave a first optimal-in-horizon polynomial regret result simultaneously with the former work, albeit using an optimism-based approach. The result of [PH20] gives a rate-optimal regret bound for unknown systems with convex costs.

In terms of **nonstochastic** control, the method-of-moments system identification procedure and the corresponding analysis presented here appeared in [HKS20], which also gave the first end-to-end regret result for nonstochastic control starting with unknown system matrices. Even earlier, [SBR19] presented a regression procedure for identifying Markov operators under nonstochastic noise. Both of these approaches have roots in the classical Ho-Kalman system identification procedure [HK66], for which the first non-asymptotic sample complexity, although for stochastic noise, was furnished in [OO19]. For strongly convex costs, logarithmic regret was established in [Sim20]. Nonstochastic control of unknown, unstable systems without any access to prior information, also called black-box control, was addressed in [CH21].

The Matrix Azuma inequality used in this chapter appears in [Tro12].

8.7 Exercises

8.1. Fix a value of T . Consider running the nonstochastic system identification procedure for $T_0 = \tilde{\Theta}(T^{2/3})$ steps to obtain system matrix estimates $\hat{A}, \hat{B}$, and subsequently using these estimates as inputs to the GPC algorithm when executed over the system (A, B) . Given Lemma 8.5, conclude that the resultant control procedure produces a regret of at most $\tilde{O}(T^{2/3})$ without any prior knowledge of system matrices, as long as the underlying system is stable.

8.2. Design an analogous control procedure as that for Exercise 8.1 with an identical regret upper bound (up to constants), when the horizon T is not known in advance.

8.3. Prove Lemma 8.4.

Part III

Learning and Filtering

Chapter 9

Learning in Unknown Linear Dynamical Systems

In previous chapters, we focused on control in dynamical systems, often assuming that the system state was fully observable and that system dynamics were either known or could be estimated through system identification. However, in many real-world scenarios, these assumptions do not hold. Instead, an agent may only have access to partial observations of the system and must learn to make accurate predictions without directly influencing the system's evolution.

This chapter focuses on learning in a dynamical system, which means predicting future system outputs based on past observations while the underlying system parameters remain unknown. Unlike control, where an agent actively influences system trajectories to minimize a given cost, learning in this setting is a passive task; the goal is to accurately forecast future outputs without modifying the system's behavior.

The ability to predict future states and outputs of an unknown system is fundamental to many domains, including time series forecasting, financial modeling, sensor networks, autonomous systems, and most recently large language models. Moreover, learning a predictive model is often a key stepping stone toward solving more complex problems, such as control under partial observability or reinforcement learning in dynamical systems. In this chapter, we study online learning algorithms for prediction in unknown linear dynamical systems, with an emphasis on regret minimization.

9.1 Learning in Dynamical Systems

The term learning in dynamical systems can refer to a variety of tasks, including state estimation, system identification, and policy learning. In this chapter, we focus on the specific task of prediction, where the objective is to estimate future outputs of a dynamical system based on past observations. Unlike classical system identification, which seeks to recover the system parameters explicitly, the learning approach taken here aims to optimize the predictions directly.

To formally define the problem, consider a sequence of observable inputs fed into a dynamical system, as illustrated in Figure 9.1. The learner observes these inputs and aims to predict the corresponding outputs while minimizing a given loss function. For the remainder of this chapter, we consider a

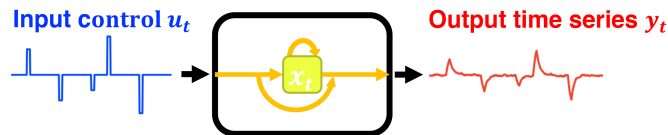


Figure 9.1: In learning we passively observe the input sequence and attempt to predict output.

partially observable linear time-invariant dynamical system described by the following equations. The extension of the material in this chapter to time-varying linear dynamical systems is left as exercise 9.6.

$$\begin{aligned}\mathbf{x}_{t+1} &= A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t \\ \mathbf{y}_t &= C\mathbf{x}_t + \mathbf{v}_t.\end{aligned}$$

Here, $\mathbf{x}_t$ represents the hidden system state, $\mathbf{u}_t$ is the observed control input, and $\mathbf{y}_t$ is the measured system output. The terms $\mathbf{w}_t$ and $\mathbf{v}_t$ account for the process noise and the observation noise, respectively. A crucial challenge in this setting is that the system matrices A, B, C are unknown to the learner.

The learner's objective is to construct an efficient prediction rule that estimates $\mathbf{y}_t$ based on past observations, despite not knowing the underlying parameters of the system.

9.2 Online Learning of Dynamical Systems

In an online learning setting, the learner sequentially observes the data and makes predictions in real time. Formally, the process unfolds as follows:

1. At each time step t , the learner observes the control input $\mathbf{u}_{t-1}$.
2. The learner makes a prediction $\hat{\mathbf{y}}_t$ for the next output.
3. The true output $\mathbf{y}_t$ is revealed.
4. The learner incurs a loss based on the difference between $\mathbf{y}_t$ and $\hat{\mathbf{y}}_t$, using a predefined loss function $\ell(\hat{\mathbf{y}}_t, \mathbf{y}_t)$, such as squared error:

$$\ell(\hat{\mathbf{y}}_t, \mathbf{y}_t) = \|\hat{\mathbf{y}}_t - \mathbf{y}_t\|^2.$$

The goal is to minimize the total loss over time. However, in adversarial or unpredictable settings, direct loss minimization may be ill-posed. Instead, we consider a regret-based formulation, where the learner's performance is measured relative to the best possible predictor from a predefined class of functions.

More precisely, let Π be a family of prediction rules mapping past observations to predictions. The regret of an online learning algorithm A with respect to Π is given by:

$$\text{Regret}_T(A, \Pi) = \sum_{t=1}^T \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2 - \min_{\pi \in \Pi} \sum_{t=1}^T \|\mathbf{y}_t - \hat{\mathbf{y}}_t^\pi\|^2$$

where $\hat{\mathbf{y}}_t^\pi$ denotes the prediction made at time t by the best function $\pi \in \Pi$ in hindsight.

This regret-based framework allows us to compare different classes of predictors and design learning algorithms that efficiently adapt to unknown dynamical systems.

9.3 Classes of Prediction Rules

We now explore what constitutes a reasonable prediction rule for the output of a linear dynamical system. However, before considering the prediction rules themselves, we discuss how to compare classes of such rules. This is analogous to the definitions and discussion in Chapter 6 where we discuss comparison between policy classes for control. However, the definitions and the corresponding conclusions for making predictions are simpler.

Definition 9.1. We say that a class of predictors Π_1 ε -approximates class Π_2 if the following holds. For every sequence of T inputs and outputs $\{\mathbf{u}_{1:T}, \mathbf{y}_{1:T}\}$, and every $\pi_2 \in \Pi_2$, there exists $\pi_1 \in \Pi_1$ such that,

$$\sum_{t=1}^T \|\mathbf{y}_t^{\pi_1} - \mathbf{y}_t^{\pi_2}\| \leq T\varepsilon,$$

where $\mathbf{y}_t^{\pi_1} = \pi_1(\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1})$, $\mathbf{y}_t^{\pi_2} = \pi_2(\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1})$

The significance of this definition is that if a simple predictor class approximates another more complicated class, then it suffices to consider only the first. The implication of ε -approximation on the regret is given below.

Lemma 9.2. Let predictor class Π_1 ε -approximate class Π_2 . Then for any prediction algorithm $\mathcal{A}$ we have

$$\text{regret}_T(\mathcal{A}, \Pi_2) \leq \text{regret}_T(\mathcal{A}, \Pi_1) + \varepsilon T.$$

The proof of this lemma is left as an exercise to the reader at the end of this chapter. We now proceed to consider specific prediction rules.

9.3.1 Linear Predictors

The simplest prediction rule is a linear function of observations, inputs, or both. More precisely,

Definition 9.3 (Linear Policies). A linear controller parametrized by matrices $\{M_{1,i}, M_{2,j}\} \in \mathbb{R}^{h \times d_y \times d_y + k \times d_y \times d_u}$ is a linear operator mapping state to control as

$$\hat{\mathbf{y}}_t = \sum_{i=1}^h M_{1,i} \mathbf{y}_{t-i} + \sum_{j=1}^k M_{2,j} \mathbf{u}_{t-j}.$$

We denote by $\Pi_{h,k,\kappa}^L$ the set of all linear controllers bounded as $\sum_{i=1}^h \|M_{1,i}\| + \sum_{j=1}^k \|M_{2,j}\| \leq \kappa$.

9.3.2 Linear Dynamical Predictors

An intuitive class of prediction rules that makes use of the special structure of linear dynamical systems is that of open-loop linear dynamical predictors, which we now define.

The motivation for this class is the case that the system transformations are known and the system is fully observed, namely $\mathbf{C} = I$, the identity

matrix. In this situation, the optimal prediction can be calculated using the system evolution equation $\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t$. It can be seen that this prediction rule is optimal for any zero-mean noise or perturbation model. However, this optimality is only true if the state is fully observable. In the next chapter, we consider optimal predictors for zero-mean i.i.d. noise in the more general partially observed case.

For a partially observed linear dynamical system without noise we can write

$$\begin{aligned} \mathbf{y}_t^\pi &= \pi_{\{A,B,C,\mathbf{x}_0\}}(t) = C\mathbf{x}_t \\ &= CA\mathbf{x}_{t-1} + CB\mathbf{u}_{t-1} \\ &\vdots \\ &= \sum_{i=1:t-1} CA^{i-1}B\mathbf{u}_{t-i} + CA^t\mathbf{x}_0. \end{aligned}$$

Thus, the operator $\pi_{\{A,B,C,\mathbf{x}_0\}}$ takes a history of states and controls and gives the next output of the system accurately. We denote by Π^* the set of all such predictors that in the case of zero noise give a perfectly accurate prediction.

If the spectral radius of A is larger than one, then the system is unstable and the magnitude of $\mathbf{y}_t$ can grow arbitrarily large, as discussed in section 4.2. Thus, for the rest of this section, assume that A is γ -stable.

With a decaying system, we can limit the memory of the predictor and still maintain accuracy. Thus, it makes sense to consider a more restricted class of predictors that have bounded memory, that is,

$$\pi_{\{A,B,C,h\}} = \sum_{i=1}^h CA^{i-1}B\mathbf{u}_{t-i},$$

where as a matter of convention we say $\mathbf{u}_t = 0$ for all negative t . Denote by Π_h^* the class of all such predictors.

Clearly, for a large h this class of predictors well approximates the true observations $\mathbf{y}_t$ if the system is activated without any noise. We next formalize what this means.

Lemma 9.4. *Suppose that $\|B\| + \|C\| \leq \gamma$, and $\|\mathbf{u}_t\| \leq U$ for all $t \in [T]$. Then the class Π_h^* for $h = \gamma \log \frac{T\gamma^3 U}{\varepsilon}$, ε -approximates the class Π^* .*

Proof. Consider any sequence and any predictor from Π^* , say $\pi_1 = \pi_{A,B,C,\mathbf{x}_0}$. Consider the predictor $\pi_2 = \pi_{A,B,C,h} \in \Pi_h^*$ for the same system matrices.

Then we have

$$\begin{aligned}
& \|\mathbf{y}_t^{\pi_1} - \mathbf{y}_t^{\pi_2}\| \\
&= \left\| \sum_{i=1:t-1} CA^{i-1}B\mathbf{u}_{t-i} + CA^{t-1}\mathbf{x}_0 - \sum_{i=1:h} CA^{i-1}B\mathbf{u}_{t-i} \right\| \\
&= \left\| \sum_{i=h}^T CA^i B\mathbf{u}_{t-i-1} \right\| \\
&\leq \sum_{i=h}^T \kappa(1-\delta)^i \|C\| \|B\| \|\mathbf{u}_{t-i-1}\| \\
&\leq \frac{\gamma^2 \kappa U}{\delta} (1-\delta)^h \leq \frac{\gamma^2 \kappa U}{\delta} e^{-\delta h},
\end{aligned}$$

for some $\frac{\kappa}{\delta} \leq \gamma$. Using the value of H , and summing up over all T iterations, we have

$$\sum_t \|\mathbf{y}_t^{\pi_1} - \mathbf{y}_t^{\pi_2}\| \leq \sum_t \frac{\gamma^2 \kappa U}{\delta} e^{-\delta h} = \varepsilon.$$

□

9.4 Efficient Learning of Linear Predictors

The previous section gives several natural classes of predictors as well as some of the relationships between them. In particular, the class of linear dynamical predictors is optimal for certain scenarios and requires knowledge of only three matrices. However, the parameterization of this class is nonconvex in the matrices A, B, C , which presents a computational challenge.

This challenge is circumvented by using the fact that the class $\prod^L \varepsilon$ -approximates the class $\prod^*$. In addition, the natural parameterization of $\prod^L$ is convex and thus can be learned using online convex optimization.

The following algorithm learns the class of linear predictors in the sense that it attains a sublinear regret bound for any sequence of adversarial cost functions, as well as perturbations and observation noise.

This algorithm is an instance of the Online Gradient Descent algorithm, and Theorem 6.1 directly gives the following bound for it. We denote by D, G the diameter of the set $\mathcal{K}$ as in Algorithm 6, and by G an upper bound on the norm of the gradients of the loss functions ℓ_t .

Algorithm 6 Learning $\prod^L$ by Online Gradient Descent

- 1: Input: $M^1 = M_{1,1:h}^1, M_{2,1:k}^1$, convex constraints set $\mathcal{K} \subseteq \mathbb{R}^{h \times d_y + k \times d_u}$
- 2: **for** $t = 1$ to T **do**
- 3: Predict $\hat{\mathbf{y}}_t(M^t) = \sum_{i=1}^h M_i^t \mathbf{u}_{t-i} + \sum_{j=1}^k M_j^t \mathbf{y}_{t-j}$ and observe true $\mathbf{y}_t$.
- 4: Suffer loss $\ell_t(\hat{\mathbf{y}}_t(M^t), \mathbf{y}_t)$, and update:

$$M^{t+1} = M^t - \eta_t \nabla \ell_t(M^t)$$

$$M^{t+1} = \Pi_{\mathcal{K}}(M^{t+1})$$

5: **end for**

Corollary 9.5. For choice of step sizes $\eta_t = \frac{D}{G\sqrt{t}}$, and $\mathcal{A}$ being Algorithm 6, we have that

$$\text{regret}(\mathcal{A}) = \sum_t \ell(\hat{\mathbf{y}}_t^{\mathcal{A}}, \mathbf{y}_t) - \min_{\pi \in \prod^L} \ell(\hat{\mathbf{y}}_t^{\pi}, \mathbf{y}_t) \leq 2GD\sqrt{T}.$$

9.5 Summary

In this chapter, we explored the problem of learning in unknown linear dynamical systems, where the goal is to predict future system outputs based on past observations while the underlying system parameters remain unknown. Unlike traditional control settings where an agent actively influences system behavior, the learning problem here is passive, focusing on accurate forecasting without intervention.

We began by formulating the problem in terms of partially observable LDS, highlighting the challenges posed by unknown system matrices and noisy observations. We discussed key approaches for online learning in dynamical systems, emphasizing regret minimization as a performance metric. The regret framework allows direct comparisons between learning algorithms by evaluating their cumulative prediction error against the best fixed predictor in hindsight.

A key takeaway from this chapter is that learning in LDS requires balancing expressivity and computational efficiency. Simple linear predictors may be computationally efficient but lack expressivity, whereas more sophisticated dynamical predictors require careful algorithmic design to remain tractable. The regret analysis provided insights into how online learning algorithms adapt to unknown system parameters while ensuring competitive performance.

This discussion sets the stage for the next chapter on Kalman filtering, where we transition from open-loop predictors to closed-loop estimation techniques that integrate feedback from observations to refine state estimates.

9.6 Bibliographic Remarks

The task of predicting future observations in an online environment is often called “filtering”. Resting at the crossroads between statistical estimation and dynamical systems, the theory of filtering had its early beginnings in the works of Norbert Wiener (see, e.g., [Wie49]).

For a comprehensive treatment of online learning in games and online convex optimization, see [CBL06, Haz16].

In the statistical noise setting, a natural approach for learning is to perform system identification and then use the identified system for prediction. This approach was taken by [SMT⁺18, SF20, SR19]. The work of [GLS⁺20] extends identification-based techniques to adversarial noise and marginally stable systems.

The class Π^* , which is the class of optimal open-loop predictors for systems with full observation, was studied in [HSZ17]. That paper showed how to learn partially observable and even marginally stable dynamical systems whose spectral radius approaches one, as long as their transition matrix is symmetric.

Under probabilistic assumptions, the work of [HMR18] shows that the matrices A, B, C, D of an unknown and partially observed time-invariant linear dynamical system can be learned using first-order optimization methods. Learning without recovery was studied in [GLS⁺20] even for marginally stable systems. More recently, tensor methods were used in [BLMY23a], and in [BLMY23b] to learn a mixture of linear dynamical systems.

9.7 Exercises

- 9.1.** Prove that a ε -approximation of prediction classes implies an additional εT regret, as formalized in the Lemma 9.2.
- 9.2.** Prove that the class Π^* is optimal for fully observed linear dynamical systems with zero mean noise.
- 9.3.** Show that the policy class Π_h^* can be learned by the policy class $\Pi_{\tilde{h},k,\kappa}^L$ with $\tilde{h} = 0$.
- 9.4.** Give precise regret bounds for learning the class $\Pi_{h,k,\kappa}^L$ and Π_h^* as a function of the natural class parameters h, k, κ and any other relevant parameter.
- 9.5.** Consider the square loss in Algorithm 6. Give an algorithm based on the Online Newton Step algorithm (see [Haz16]) and write down the resulting regret bound.
- 9.6.** Write down the predictor class Π^* and Π_h^* for time-varying linear dynamical systems. Research the notion of adaptive regret from [Haz16], and reason about the conditions under which the class Π^L approximates Π^* for time-varying linear dynamical systems.

Chapter 10

Kalman Filtering

State estimation plays a central role in control and prediction tasks, where an agent seeks to infer latent states from noisy observations. The Kalman filter is a classical and widely used method for solving this problem, providing optimal state estimates under the assumption of linear system dynamics and Gaussian noise.

A key feature of the Kalman filter and a fundamental limitation is that it requires knowledge of the system matrices, including both the state transition matrix and the observation model. These matrices encode how the system evolves over time and how observations relate to the hidden state. Although the Kalman filter is provably optimal when these parameters are known, real-world applications often involve partially known or entirely unknown system dynamics, making explicit identification challenging.

In contrast, online learning approaches, such as those introduced in Chapter 9, provide a model-free open-loop alternative by directly learning optimal linear predictors from the data. Instead of estimating a latent state and system matrices, these methods construct forecasting models using past observations, making them more adaptive to unknown or time-varying systems. However, these approaches do not incorporate feedback, meaning that they operate in an open-loop setting, predicting future states without correcting for new observations.

This chapter moves beyond open-loop prediction to closed-loop state estimation, where past observations continuously refine future predictions. The Kalman filter accomplishes this by recursively updating estimates based on incoming measurements, achieving Bayes-optimal performance under Gaussian noise. This structured feedback mechanism allows for more accurate and

robust estimation compared to open-loop methods, particularly in dynamic and uncertain environments.

Beyond classical Kalman filtering, we also introduce a closed-loop online learning method that serves as a convex relaxation of the Kalman filter. This approach retains the Kalman filtering feedback mechanism, while eliminating the need for explicit system identification, making it applicable in adversarial or uncertain environments. By formulating state estimation as an online convex optimization problem, this method provides provable regret guarantees and serves as a bridge between Bayesian filtering and learning-based estimation techniques.

We provide a rigorous introduction to Kalman filtering, including its derivation, optimality properties, and computational aspects. We also discuss its limitations and contrast it with learning-based alternatives, setting the stage for spectral filtering techniques in the next chapter, which further generalize state estimation in unknown dynamical systems.

10.1 Observable Systems

Observability is a fundamental concept in control theory that determines whether the internal state of a system can be uniquely inferred from its output over time. In the infinite-horizon setting, prediction is meaningful only if observations remain bounded, a condition closely related to system controllability (as discussed in Section 4.3).

A general dynamical system maps an initial state to an infinite sequence of observations. If this mapping is injective, meaning that each initial state corresponds to a unique sequence of observations, then the system is said to be observable. Formally:

Definition 10.1. *A partially observed dynamical system is observable if and only if every sequence of observations uniquely determines the initial hidden state.*

A simple example of an unobservable system is given as the following variant of the double integrator from section 2.1.2. Consider the double integrator dynamical system with $\Delta = 1$ and partial observability where the observation matrix is given as follows

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad C = [0 \ 1].$$

This system allows only the velocity of the object to be observed, while the position remains hidden. It can be seen that this setting is unobservable,

as the initial position cannot be determined on the basis of the velocities and forces applied to the system alone. Indeed, this is a special case of the Galilean Relativity principle of Newtonian mechanics, which states that the laws of physics are the same in all inertial reference frames.

For linear systems, observability can be verified using a rank condition on the Kalman observability matrix, given by:

$$\hat{K}(A, C) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{d_{\mathbf{x}}-1} \end{bmatrix}.$$

where A and C are the system matrices that define the state transition and observation model, respectively. Notice that in our example, the Kalman observability matrix is given by $\hat{K}(A, C) = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$, which is degenerate. This is not by chance; the system is observable if and only if the observability matrix has full rank. We ask the reader to provide a proof of the following result in Exercise 10.1.

Theorem 10.2. *A partially-observed dynamical system specified by system parameters (A, B, C) is observable if and only if $\text{rank}(\hat{K}(A, C)) = d_{\mathbf{x}}$.*

10.2 Optimality of the Kalman Filter

10.2.1 The Power of Linear Predictors

One of the important conclusions from this chapter is that the policy class of linear predictors is optimal with respect to closed-loop policies. We prove this by proving two facts:

1. The class of linear policies ε -approximates the class of Kalman filters, in the sense of Definition 9.1.
2. The class of Kalman filters is optimal for closed-loop policies.

The conclusion from these two facts is that algorithm 6 can be used to learn a policy that competes with the best closed-loop optimal predictor in hindsight. If the signal comes from a linear dynamical system subject to Gaussian noise, this means competing with the optimal predictor, which is a Kalman filter. We prove this formally at the end of this section.

10.2.2 Closed-Loop Optimal Predictors

Open-loop predictors, such as those introduced in the previous chapter, leverage system inputs and initial conditions to make predictions. However, they do not incorporate feedback from observations, making them susceptible to modeling errors and disturbances.

Closed-loop prediction rules address this limitation by continuously adjusting estimates based on new observations. The most well-known example of such a predictor is the Kalman filter, which produces the best linear predictor in a least-squares sense under the assumption that process and observation noise are Gaussian with known covariances.

Formally, let $\mathbf{w}_t$ and $\mathbf{v}_t$ denote the process and observation noise, assumed to be zero-mean and i.i.d., with covariance matrices:

$$\Sigma_x \stackrel{\text{def}}{=} \mathbb{E} \left[\mathbf{w}_t \mathbf{w}_t^\top \right], \quad \Sigma_y = \mathbb{E} \left[\mathbf{v}_t \mathbf{v}_t^\top \right].$$

So far, we have focused on predicting observations, that is, $\mathbf{y}_t$. However, the Kalman filter also provides optimal state estimates $\hat{\mathbf{x}}_t$ given past observations $\mathbf{y}_{1:t}$ and controls $\mathbf{u}_{1:t}$. This is motivated by the fact that the least-squares optimal predictor of observations can be expressed as:

$$\mathbf{y}_t = C\mathbf{x}_t.$$

In Exercise 10.6, we ask the reader to provide a proof of this statement.

Based upon this observation, the following theorem is the essence of the Kalman filter, which recovers estimates for states as well as linear projection matrices for optimal prediction of the next observation.

Theorem 10.3. *Let the control sequence $\mathbf{u}_t$ be an arbitrary affine function of $\mathbf{y}_{1:t}$. There exists an efficiently computable sequence of linear predictors $\hat{\mathbf{x}}_{t+1}$ defined recursively as*

$$\hat{\mathbf{x}}_{t+1} = (A - L_t C)\hat{\mathbf{x}}_t + B\mathbf{u}_t + L_t \mathbf{y}_t,$$

such that

$$\mathbb{E} \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|^2 \leq \min_{L \in \Pi^L} \mathbb{E} \|\mathbf{x}_t - L[\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1}]\|^2.$$

To prove this theorem, we require some basic tools from linear regression which we state next.

10.2.3 Tools from Linear Regression

We begin with an observation about least-squares predictors, which we ask the reader to prove in Exercise 10.3.

Lemma 10.4. *Let X, Y, Z be matrix-valued random variables. Let*

$$A^* = \arg \min_A \mathbb{E} \|X - AY\|^2, \text{ and } B^* = \arg \min_B \mathbb{E} \|X - BZ\|^2.$$

If $\mathbb{E}[YZ^\top] = 0$, then

$$\mathbb{E} \|X - A^*Y - B^*Z\|^2 \leq \min_{A,B} \mathbb{E} \|X - AY - BZ\|^2.$$

In the remaining proof, we use this observation to construct an optimal least-squares estimator $\hat{\mathbf{x}}_{t+1}$ for $\mathbf{x}_{t+1}$ given $\hat{\mathbf{x}}_t$. In particular, we know by induction hypothesis that

$$\mathbb{E} \|\mathbf{x}_t - \hat{\mathbf{x}}_t\|^2 \leq \min_{L \in \Pi^L} \mathbb{E} \|\mathbf{x}_t - L[\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1}]\|^2.$$

But, is $\hat{\mathbf{x}}_t$ any good for predicting $\mathbf{x}_{t+1}$? To answer this, we note the following structural characterization of least-squares (Exercise 10.2).

Lemma 10.5. *Let X, Y be possibly correlated random variables. Then*

$$A^* \stackrel{\text{def}}{=} \arg \min_A \mathbb{E} \|X - AY\|^2 = \mathbb{E} [XY^\top] \left(\mathbb{E} [YY^\top] \right)^{-1}.$$

*Furthermore, if $\hat{X} = A^*Y$, then $\mathbb{E}(X - \hat{X})Y^\top = 0$.*

10.2.4 Analysis

We are now ready to prove the main property of the Kalman filter below.

Proof of Theorem 10.3. Since the estimator is defined recursively, a natural strategy is to proceed by induction. Let us say $\hat{\mathbf{x}}_t$ is such a predictor. Furthermore, let

$$\Sigma_t \stackrel{\text{def}}{=} \mathbb{E}(\mathbf{x}_t - \hat{\mathbf{x}}_t)(\mathbf{x}_t - \hat{\mathbf{x}}_t)^\top.$$

Since for any $s < t$

$$\mathbb{E} [(\mathbf{x}_{t+1} - B\mathbf{u}_t)\mathbf{y}_s^\top] = \mathbb{E} [(A\mathbf{x}_t + \mathbf{w}_t)\mathbf{y}_s^\top] = A\mathbb{E} [\mathbf{x}_t\mathbf{y}_s^\top],$$

using the first part of Lemma 10.5, we have that

$$\mathbb{E}\|\mathbf{x}_{t+1} - A\hat{\mathbf{x}}_t - B\mathbf{u}_t\|^2 \leq \min_{L \in \Pi^L} \mathbb{E}\|\mathbf{x}_{t+1} - B\mathbf{u}_t - L[\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1}]\|^2.$$

But, when predicting $\mathbf{x}_{t+1}$, we have more information. In particular, we also have $\mathbf{y}_t$. Consider random variables $\mathbf{y}_t - C\hat{\mathbf{x}}_t$. Can we use this covariate alone to predict $\mathbf{x}_{t+1}$? Note that

$$\begin{aligned} \mathbb{E}\left[(\mathbf{y}_t - C\hat{\mathbf{x}}_t)(\mathbf{y}_t - C\hat{\mathbf{x}}_t)^\top\right] &= \Sigma_v + C\mathbb{E}\left[(\mathbf{x}_t - \hat{\mathbf{x}}_t)(\mathbf{x}_t - \hat{\mathbf{x}}_t)^\top\right]C^\top \\ &= \Sigma_v + C\Sigma_t C^\top, \end{aligned}$$

$$\begin{aligned} \mathbb{E}\left[(\mathbf{x}_{t+1} - B\mathbf{u}_t)(\mathbf{y}_t - C\hat{\mathbf{x}}_t)^\top\right] &= \mathbb{E}\left[(A\mathbf{x}_t + \mathbf{w}_t)(C(\mathbf{x}_t - \hat{\mathbf{x}}_t) + \mathbf{v}_t)^\top\right] \\ &= A\mathbb{E}\left[(\mathbf{x}_t - \hat{\mathbf{x}}_t)(\mathbf{x}_t - \hat{\mathbf{x}}_t)^\top\right]C^\top + A\underbrace{\mathbb{E}\left[\hat{\mathbf{x}}_t(\mathbf{x}_t - \hat{\mathbf{x}}_t)^\top\right]}_{=0}C^\top \\ &= A\Sigma_t C^\top, \end{aligned}$$

where the equality indicated in braces follows from the second part of Lemma 10.5, by noting that $\hat{\mathbf{x}}_t$ is a linear function of $\mathbf{y}_{1:t-1}$.

In the preceding paragraphs, we have constructed two least-squares predictors for $\hat{\mathbf{x}}_{t+1}$. Observe that for any $s < t$

$$\mathbb{E}(\mathbf{y}_t - C\hat{\mathbf{x}}_t)\mathbf{y}_s^\top = C\mathbb{E}(\mathbf{x}_t - \hat{\mathbf{x}}_t)\mathbf{y}_s^\top = 0,$$

where the last equality once again follows from Lemma 10.5. This orthogonality allows us to invoke Lemma 10.4 to arrive at

$$\hat{\mathbf{x}}_{t+1} = A\hat{\mathbf{x}}_t + B\mathbf{u}_t + \underbrace{A\Sigma_t C^\top (\Sigma_y + C\Sigma_t C^\top)^{-1}}_{\stackrel{\text{def}}{=} L_t} (\mathbf{y}_t - C\hat{\mathbf{x}}_t),$$

where the last component follows from Lemma 10.5 and the correlations we just computed.

To establish that such predictors can be computed efficiently, note that

$$\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1} = (A - L_t C)(\mathbf{x}_t - \hat{\mathbf{x}}_t) + \mathbf{w}_t + L_t \mathbf{v}_t.$$

Finally, we give a recursive relation on Σ_t as follows—

$$\begin{aligned} \Sigma_{t+1} &\stackrel{\text{def}}{=} \mathbb{E}\left[(\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1})(\mathbf{x}_{t+1} - \hat{\mathbf{x}}_{t+1})^\top\right] \\ &= (A - L_t C)\Sigma_t(A - L_t C)^\top + \Sigma_x + L_t \Sigma_y L_t^\top \\ &= A\Sigma_t A^\top + \Sigma_x + L_t(C\Sigma_t C^\top + \Sigma_y)L_t^\top - 2A\Sigma_t C^\top L_t^\top \\ &= A\Sigma_t A^\top - A\Sigma C^\top(C\Sigma_t C^\top + \Sigma_y)^{-1}C\Sigma A^\top + \Sigma_x. \end{aligned}$$

□

10.2.5 Infinite Horizon Kalman Filtering

In the infinite-horizon setting, the steady-state Kalman filter can be described by the following linear dynamical system

$$\begin{aligned}\hat{\mathbf{x}}_{t+1} &= (A - LC)\hat{\mathbf{x}}_t + B\mathbf{u}_t + L\mathbf{y}_t, \\ \hat{\mathbf{y}}_t &= C\hat{\mathbf{x}}_t,\end{aligned}$$

where $L = A\Sigma(C\Sigma C^\top + \Sigma_y)^{-1}$ and Σ satisfies the Riccati equation

$$\Sigma = A\Sigma A^\top - A\Sigma C^\top (C\Sigma C^\top + \Sigma_y)^{-1} C\Sigma A^\top + \Sigma_x.$$

However, when does a solution to this Riccati equation exist? A sufficient condition for this is the observability of the underlying dynamical system. In particular, any observable partially-observed linear dynamical system admits a solution to the Riccati equation listed above, and hence, has a steady-state Kalman filter. Mirroring the link between controllability and stabilizability, observability implies $\rho(A - LC) < 1$.

10.2.6 Linear Predictors and the Kalman Filter

For observable linear dynamical systems, the Kalman filter is contained in the class of linear predictors. This is formally stated in the following lemma, whose proof is left as Exercise 10.4.

Lemma 10.6. *Consider any partially-observed linear dynamical system (A, B, C) where the Kalman filtering matrix L is such that the system $A - LC$ is γ -stable. The Kalman filtering policy is ε -contained in the class of linear policies $\Pi_{h,k,\kappa}^L$, where $h = k = \gamma \log \frac{\gamma T}{\varepsilon}$ and $\kappa = \gamma^3$.*

10.3 Bayes-Optimality under Gaussian Noise

In this subsection, we further restrict $\mathbf{w}_t, \mathbf{v}_t$ to be i.i.d. draws from Gaussian distributions, and proceed to establish that the Kalman filter in such cases enjoys a stronger notion of optimality; it produces the best estimator (under a least-squares loss) of the state given past observations among the class of all (not just linear) predictors.¹ The setting we consider is that of

¹This is also called the Bayes-optimal predictor. See bibliographic materials for more details and literature.

Gaussian perturbations $\mathbf{w}_t$ and $\mathbf{v}_t$ and known systems matrices (A, B, C) . In particular, we consider the case where $\mathbf{w}_t \sim \mathcal{N}(0, \Sigma_x)$ and $\mathbf{v}_t \sim \mathcal{N}(0, \Sigma_y)$ are i.i.d. Gaussian and $x_0 = 0$. Given the realizations $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_t$ under some control input sequence $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_t$, we would like to compute a prediction $\hat{\mathbf{y}}_{t+1}$ that minimizes the mean square error, which is simply the expected observation as per

$$\hat{\mathbf{y}}_{t+1} = \mathbb{E}[\mathbf{y}_{t+1} | \mathbf{y}_{1:t}, \mathbf{u}_{1:t}].$$

Since $\hat{\mathbf{y}}_{t+1} = C\hat{\mathbf{x}}_{t+1}$, we will recursively compute $\hat{\mathbf{x}}_{t+1}$ where

$$\hat{\mathbf{x}}_{t+1} = \mathbb{E}[\mathbf{x}_{t+1} | \mathbf{y}_{1:t}, \mathbf{u}_{1:t}].$$

Theorem 10.7. *The conditional expectation of the state $\hat{\mathbf{x}}_{t+1}$ satisfies*

$$\hat{\mathbf{x}}_{t+1} = (A - L_t C)\hat{\mathbf{x}}_t + B\mathbf{u}_t + L_t \mathbf{y}_t,$$

for a sequence of matrices $L_t \in \mathbb{R}^{d_x \times d_y}$ that can be efficiently computed given (A, B, C) , and is independent of the realizations $\mathbf{u}_{1:T}, \mathbf{y}_{1:T}$.

Proof. From the Gaussian nature of perturbations, we observe that

$$\begin{bmatrix} \mathbf{x}_t | (\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1}) \\ \mathbf{y}_t | (\mathbf{y}_{1:t-1}, \mathbf{u}_{1:t-1}) \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \hat{\mathbf{x}}_t \\ C\hat{\mathbf{x}}_t \end{bmatrix}, \begin{bmatrix} \Sigma_t & \Sigma_t C^\top \\ C\Sigma_t & C\Sigma_t C^\top + \Sigma_y \end{bmatrix} \right).$$

for some $\Sigma_t \in \mathbb{R}^{d_x \times d_x}$, for which we provide a recursive formula towards the end of the proof. We make note of the following lemma for Gaussian distributions, whose proof we ask the reader to complete in Exercise 10.5.

Lemma 10.8. *If a random variable pair $(\mathbf{x}, \mathbf{y})$ are distributed as*

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right),$$

then $\mathbf{x} | \mathbf{y} \sim \mathcal{N}(\mu_x + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y} - \mu_y), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$.

Therefore, we have that

$$\begin{aligned} \mathbf{x}_t | (\mathbf{y}_{1:t}, \mathbf{u}_{1:t-1}) &\sim \mathcal{N}(\hat{\mathbf{x}}_t + \Sigma_t(C\Sigma_t C^\top + \Sigma_y)^{-1}(\mathbf{y}_t - C\hat{\mathbf{x}}_t), \Sigma'_t), \\ \Sigma'_t &= \Sigma_t - \Sigma_t C^\top (C\Sigma_t C^\top + \Sigma_y)^{-1} C\Sigma_t. \end{aligned}$$

Now, using this, we arrive at

$$\begin{aligned} \hat{\mathbf{x}}_{t+1} &= A\mathbb{E}[\mathbf{x}_t | \mathbf{y}_{1:t}, \mathbf{u}_{1:t-1}] + B\mathbf{u}_t \\ &= (A - \underbrace{A\Sigma_t(C\Sigma_t C^\top + \Sigma_y)^{-1}C}_{\stackrel{\text{def}}{=} L_t})\hat{\mathbf{x}}_t + B\mathbf{u}_t + \underbrace{A\Sigma_t(C\Sigma_t C^\top + \Sigma_y)^{-1}\mathbf{y}_t}_{\stackrel{\text{def}}{=} L_t \mathbf{y}_t}. \end{aligned}$$

It remains to obtain a recursive relation on Σ_t . To this extent, from the state evolution of a linear dynamical system, we have that

$$\Sigma_{t+1} = A\Sigma'_tA^\top + \Sigma_x = A\Sigma_tA^\top - A\Sigma_tC^\top(C\Sigma_tC^\top + \Sigma_y)^{-1}C\Sigma_tA^\top + \Sigma_x.$$

□

10.4 Conclusion

In this chapter, we explored the Kalman filter as a fundamental tool for state estimation in dynamical systems, particularly in the presence of noise and uncertainty. We established its optimality in the least-squares sense and examined its connections to online convex optimization, reinforcing its relevance in both classical control and modern learning-based paradigms.

Key insights from this chapter include the role of observability in ensuring accurate state estimation, the derivation of the Kalman filter update equations, and the recursive structure that allows for efficient real-time implementation. We also presented the filter's steady-state behavior, particularly in the infinite-horizon setting, where it converges to a solution governed by the Riccati equation.

Beyond its theoretical guarantees, the Kalman filter is widely used in practical applications ranging from autonomous systems and robotics to finance and neuroscience. However, challenges remain, particularly in scenarios with non-Gaussian noise, time-varying dynamics, or adversarial disturbances. These limitations motivate extensions such as robust filtering, nonlinear filtering (e.g., the extended and unscented Kalman filters) and learning-based adaptations that integrate statistical inference with modern machine learning techniques.

The next chapter will transition from the classical filtering framework to spectral filtering, providing an alternative approach to state estimation that leverages spectral methods for efficient learning in dynamical systems.

10.5 Bibliographic Remarks

In 1960, Rudolph Kalman gave a state-space solution to the filtering problem that was highly amenable to further extensions [Kal60]. See [AM91] for a historical perspective on its development and aftermaths.

For unknown systems, in the stochastic case, one may estimate the system matrices, as we have discussed before, before applying optimal filtering. Improper learning approaches to this problem were studied in [KMTM19].

State estimation is a fundamental problem in control and signal processing, with numerous approaches beyond the classical Kalman filter. Alternative methods include:

- **Nonlinear Filtering Extensions.** The Kalman filter assumes that the dynamics is linear system and the noise is Gaussian, which limits its effectiveness in nonlinear settings. The Extended Kalman Filter (see, for example, [UA21]) addresses this issue by linearizing the dynamics at each time step via a first-order Taylor series expansion. However, linearization can introduce significant errors in highly nonlinear systems. Notable extensions using more sophisticated sampling include the Unscented Kalman Filter [JU04], leading to better estimation accuracy in strongly nonlinear systems.
- **Particle Filtering (Sequential Monte Carlo Methods)**
For highly nonlinear and non-Gaussian settings, particle filters [GSS93] provide a flexible alternative. These methods approximate the posterior distribution using a set of weighted samples (particles) and employ importance sampling and resampling techniques.
- **Adversarial & Robust Filtering**
Robust filtering methods seek to provide guarantees under adversarial noise. The H_∞ filter [BB08] minimizes the worst-case estimation error rather than the mean-squared error, making it suitable for applications with unknown disturbances.

Although the Kalman filter remains a cornerstone of state estimation, these alternative methods provide valuable extensions and improvements for nonlinear, adversarial, and data-driven settings. In the next chapter, we will study an alternative using a technique which circumvents the need for state estimation altogether and has stronger guarantees for sequential prediction.

10.6 Exercises

10.1. In this exercise we prove Theorem 10.2 in two parts:

1. Show that if the rank of the Kalman observability matrix is less than $d_{\mathbf{x}}$, then there are two initial states that after $d_{\mathbf{x}}$ iterations map to the same state. Hence, the initial state is not uniquely determined.
2. In contrast, show that if the rank of $\hat{K}$ is full, then at any time t , $\mathbf{x}_t$ determines $\mathbf{x}_0$ uniquely.

10.2. Prove Lemma 10.5, which states the following. Let X, Y be possibly correlated random variables. Then

$$A^* \stackrel{\text{def}}{=} \arg \min_A \mathbb{E} \|X - AY\|^2 = \mathbb{E} [XY^\top] (\mathbb{E} [YY^\top])^{-1}.$$

Furthermore, if $\hat{X} = A^*Y$, then $\mathbb{E}(X - \hat{X})Y^\top = 0$.

10.3. Using Lemma 10.5, prove Lemma 10.4, which states the following. Let X, Y, Z be matrix-valued random variables. Let

$$A^* = \arg \min_A \mathbb{E} \|X - AY\|^2, \text{ and } B^* = \arg \min_B \mathbb{E} \|X - BZ\|^2.$$

If $\mathbb{E} [YZ^\top] = 0$, then

$$\mathbb{E} \|X - A^*Y - B^*Z\|^2 \leq \min_{A, B} \mathbb{E} \|X - AY - BZ\|^2.$$

10.4. In this exercise we conclude that the Kalman filter can be improperly learned by the class of linear predictors. Prove this fact as formally stated in Lemma 10.6. Conclude that the Kalman filter can be learned using Algorithm 6.

10.5. Prove Lemma 10.8, which is stated below. If a random variable pair $(\mathbf{x}, \mathbf{y})$ are distributed as

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right),$$

then the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ can be written as $\mathbf{x}|\mathbf{y} \sim \mathcal{N}(\mu_x + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y} - \mu_y), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$.

10.6. Using Lemma 10.5, prove that the optimal least-squares estimator for the observation at time t given past observations $y_{1:t-1}$ and control inputs $u_{1:t-1}$ is given by $\hat{\mathbf{y}}_t = C\hat{\mathbf{x}}_t$, where $\hat{\mathbf{x}}_t$ is as defined in Theorem 10.3.

Chapter 11

Spectral Filtering

In the previous chapter, we explored two major approaches for state estimation and prediction in unknown linear dynamical systems:

1. State estimation via Kalman filtering, which recursively estimates both the latent system state and the projection matrices that map the system state to observations. Kalman filtering provides an optimal solution under Gaussian noise, and updates the state estimates over time. However, it requires an accurate model of system dynamics, including transition and observation matrices, which may be difficult to estimate, especially in adversarial or high-noise environments.
2. Online learning of linear predictors, which offers a relaxation of the Kalman filtering approach by directly learning predictive models without explicit system identification. Instead of estimating the full state and transition matrices, this approach constructs a forecasting model using past inputs and outputs. Although this avoids the complexity of full system identification, it introduces a major limitation: the number of parameters required depends on the stability constant of the system, which can be large and unknown, making learning impractical for unstable or poorly conditioned systems.

In this chapter, we introduce Spectral Filtering, an alternative approach that bypasses these limitations. Rather than relying on state estimation or direct system identification, spectral filtering constructs predictors on a transformed basis, which does not depend on the stability properties of the system. This method leverages spectral decompositions to efficiently extract the relevant dynamical structure, enabling more robust and scalable learning in both stable and unstable systems.

To illustrate the power of spectral filtering, we first present a one-dimensional case, which provides an intuitive understanding of how spectral methods can be applied to system identification and forecasting. We then extend the discussion to higher-dimensional settings, demonstrating how spectral predictors enable a structured and computationally efficient representation of system dynamics.

Although this chapter primarily focuses on symmetric system matrices for clarity, the principles of spectral filtering extend to asymmetric and more general linear dynamical system settings. These extensions require additional mathematical tools, which we reference in the bibliographic section.

11.1 Spectral filtering in One Dimension

In this section, we give a simple introduction to the technique of spectral filtering, restricting ourselves to a single dimension. The restriction to a single dimension simplifies the exposition and is yet sufficient to illustrate the main ideas. In the next section, we consider the high-dimensional case.

The one-dimensional systems which we consider next are scalar systems, with $d_u = d_y = d_x = 1$. For simplicity, we assume that the initial state is zero $\mathbf{x}_0 = 0$. Recall the set of optimal predictors, the class $\Pi_\star$, which for scalar systems can be written and simplified to,

$$\mathbf{y}_t = \sum_{i=1}^{t-1} C A^{i-1} B \mathbf{u}_{t-i} = \beta \sum_{i=1}^{t-1} \alpha^{i-1} \mathbf{u}_{t-i} = \beta \cdot \tilde{\mathbf{u}}_t^\top \mu_\alpha, \quad (11.1)$$

where $\beta = CB \in \mathbb{R}$ is a scalar and we used the vector notation for

$$\mu_\alpha = [1 \quad \alpha \quad \alpha^2 \quad \dots \quad \alpha^{T-1}]^\top \in \mathbb{R}^T,$$

and we define $\tilde{\mathbf{u}}_t$ to be the padded vector $\mathbf{u}_{t-1:1}$:

$$\tilde{\mathbf{u}}_t = [\mathbf{u}_{t-1} \quad \dots \quad \mathbf{u}_1 \quad 0 \quad \dots \quad 0]^\top \in \mathbb{R}^T.$$

The main idea of spectral filtering is to rewrite the vectors μ_α in a more efficient basis, as opposed to the standard basis. The notion of efficiency comes from the definition of ε -approximation that we have studied in previous chapters. To define the basis, we consider the following matrix:

$$Z_T = \int_0^1 \mu_\alpha \mu_\alpha^\top d\alpha \in \mathbb{R}^{T \times T}.$$

The matrix Z_T is symmetric and is independent of α . Let $\phi_1, \dots, \phi_h, \dots \in \mathbb{R}^T$ be the eigenvectors of Z_T ordered by the magnitude of their corresponding eigenvalues. Our basis for representing μ_α is spanned by the eigenvectors of Z_T , and we proceed to study their properties.

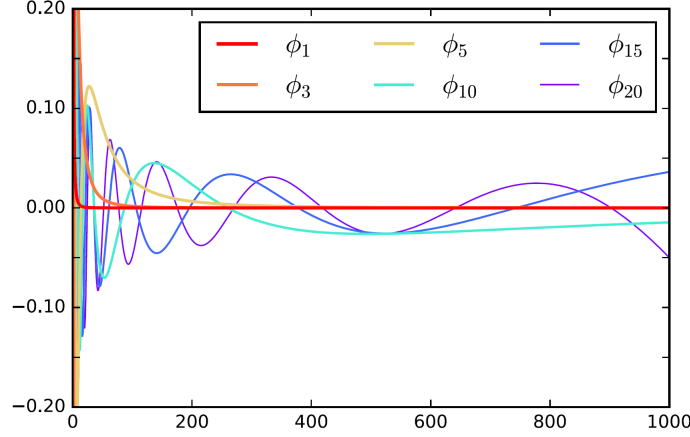


Figure 11.1: The filters obtained by the eigenvectors of Z_T .

11.1.1 The Magic of Hankel Matrices

The matrix Z_T is called the Hilbert matrix, and it is a special case of a Hankel Matrix, which is a square Hermitian matrix with each ascending skew-diagonal from left to right having the same value. To spell out Z_T more precisely,

$$Z_T = \int_0^1 \begin{bmatrix} 1 & \alpha & \alpha^2 & & \\ \alpha & \alpha^2 & \alpha^3 & & \\ \alpha^2 & \alpha^3 & \alpha^4 & & \\ & & & \dots & \\ & & & & \alpha^{2T-2} \end{bmatrix} d\alpha = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & & \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & & \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & & \\ & & & \dots & \\ & & & & \frac{1}{2T-1} \end{bmatrix}.$$

The properties of Hankel matrices have been studied extensively, and the following theorem captures one of their crucial properties.

Theorem 11.1. *Let σ_k be the k -th largest singular value of the Hilbert matrix $Z_T \in \mathbb{R}^{T \times T}$, then for $k, T > 1$:*

$$\sigma_k(Z_T) \leq 4\pi^2 \cdot e^{-\frac{k}{\log T}}$$

The proof of this theorem is beyond our scope, and references to details are given in the bibliographic material at the end of this chapter.

This theorem shows that the eigenvalues of a Hankel matrix decay very rapidly. An additional illustration of this fact is given in Figure 11.2, which plots the eigenvalues of the matrix Z_T for $T = 100$ on a logarithmic scale.

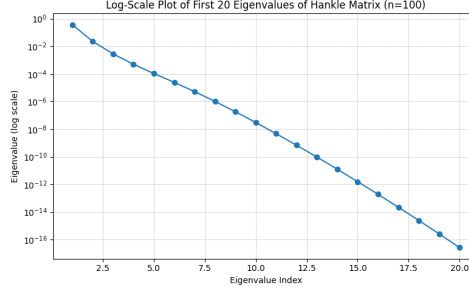


Figure 11.2: Eigenvalues of Hankel matrices decrease geometrically. These are the eigenvalues of the matrix Z_T , where $T = 100$, plotted on a logarithmic scale.

11.2 Spectral Predictors

The basis of eigenvectors of a Hankel matrix gives rise to an interesting class of predictors. This class of mappings does not directly apply to the history of inputs and/or observations. Instead, we first apply a linear transformation over these vectors, and then apply a linear map. More formally,

Definition 11.2. Define the class of spectral linear predictors of order h as

$$\Pi_h = \left\{ \pi_M \mid \pi_M(\mathbf{u}_{1:t}, \mathbf{y}_{1:t}) = \sum_{i=1}^h M_i(\tilde{\mathbf{u}}_t^\top \phi_i) \right\}.$$

Here $\tilde{\mathbf{u}}_t^\top \phi_i$ evaluates the inner product of the i 'th eigenvector of Z_T with the padded history vector $\tilde{\mathbf{u}}_t$, where $\tilde{\mathbf{u}}_t = [\mathbf{u}_{t-1} \ \dots \ \mathbf{u}_1 \ 0 \ \dots \ 0]^\top \in \mathbb{R}^T$.

Notice that as $h \rightarrow T$, this class is equivalent to the class of all linear predictors on the inputs Π^{LIN} . The reader may wonder at this point: Since a linear transformation composed with another linear transform remains linear, what is the benefit here?

The answer is that linear transformations are equivalent when we have full-rank transforms. In our case, however, we consider low-rank approximations to the transform, and the approximation properties of different subspaces can differ drastically! This is the case in our setting: the spectral basis arising from the matrix Z_T gives excellent approximations to the class of (standard) linear predictors, as we formally prove next. The following lemma is analogous to Lemma 9.4, with the significant difference that the number of parameters h does not depend on the stabilizability parameter of the system γ at all!

Lemma 11.3. *Suppose that $\mathbf{x}_0 = 0$ and $|u_t| \leq 1$ for all t . Then the class Π_h for $h = O(\log^2 \frac{T}{\varepsilon})$ ε -approximates the class $\Pi_\star$.*

The proof of this lemma relies on the following crucial property of the spectral basis.

Lemma 11.4. *Let $\{\phi_j\}$ be the orthonormal eigenvectors of Z_T , with corresponding eigenvalues σ_j . Then for all $j \in [T]$, $j, T > 1$, and $\alpha \in [0, 1]$, we have*

$$|\phi_j^\top \mu_\alpha| \leq (4T^2 \sigma_j)^{1/4}.$$

In particular, using Theorem 11.1,

$$|\phi_j^\top \mu_\alpha| = O\left(T^{1/2} e^{-\frac{j}{4 \log T}}\right).$$

Using this property, we can prove Lemma 11.3 by shifting to a representation on the spectral basis as follows.

Proof of Lemma 11.3. Consider any sequence and any predictor from $\Pi_\star$, say $\pi_1 = \pi_{A,B,C,0}$. Following equation (11.1), we can write

$$\begin{aligned} \mathbf{y}_t^{\pi_1} &= \beta \cdot \tilde{\mathbf{u}}_t^\top \mu_\alpha \\ &= \beta \cdot \tilde{\mathbf{u}}_t^\top \left(\sum_{i=1}^T \phi_i \phi_i^\top \right) \mu_\alpha \quad (\text{basis property } \sum_{i=1}^T \phi_i \phi_i^\top = I) \\ &= \sum_{i=1}^T M_i^* (\tilde{\mathbf{u}}_t^\top \phi_i), \quad M_i^* = \beta (\phi_i^\top \mu_\alpha). \end{aligned}$$

By Lemma 11.4,

$$\begin{aligned} |M_i^*| &= |\beta| |\phi_i^\top \mu_\alpha| \leq |\beta| (4T^2 \sigma_i)^{1/4} \\ &= O\left(|\beta| T^{1/2} e^{-\frac{i}{4 \log T}}\right). \end{aligned} \tag{11.2}$$

Consider the predictor $\pi_2 \in \Pi_h$ that keeps only the first h coefficients. Then we have

$$\begin{aligned}
|\mathbf{y}_t^{\pi_1} - \mathbf{y}_t^{\pi_2}| &= \left| \sum_{i=1}^T M_i^* (\tilde{\mathbf{u}}_t^\top \phi_i) - \sum_{i=1}^h M_i^* (\tilde{\mathbf{u}}_t^\top \phi_i) \right| \\
&= \left| \sum_{i=h+1}^T M_i^* (\tilde{\mathbf{u}}_t^\top \phi_i) \right| \\
&\leq \sum_{i=h+1}^T |M_i^*| |\tilde{\mathbf{u}}_t^\top \phi_i| \\
&\leq \sum_{i=h+1}^T |M_i^*| \|\phi_i\|_2 \|\tilde{\mathbf{u}}_t\|_2 \quad (\text{Cauchy-Schwarz}) \\
&\leq \sqrt{T} \sum_{i=h+1}^T |M_i^*| \quad \|\phi_i\|_2 = 1, |\mathbf{u}_t| \leq 1 \\
&\leq \sqrt{T} |\beta| T^{1/2} \sum_{i=h+1}^\infty e^{-\frac{i}{4 \log T}} \quad (11.2)
\end{aligned}$$

The geometric tail can be strictly bounded via an integral:

$$\sum_{i=h+1}^\infty e^{-\frac{i}{4 \log T}} \leq \int_h^\infty e^{-\frac{x}{4 \log T}} dx = 4 \log T \cdot e^{-\frac{h}{4 \log T}}.$$

Thus, the overall error is bounded by:

$$|\mathbf{y}_t^{\pi_1} - \mathbf{y}_t^{\pi_2}| \leq |\beta| T (4 \log T) e^{-\frac{h}{4 \log T}}.$$

Choosing

$$h \geq 4 \log T \left(\log T + \log(4 \log T) + \log \frac{|\beta|}{\varepsilon} \right)$$

ensures the per-round error is at most ε , completing the proof. $\square$

Proof of Lemma 11.4. Consider the scalar function

$$g(\alpha) = (\mu_\alpha^\top \phi_j)^2$$

over the interval $[0, 1]$.

First,

$$\begin{aligned}
\int_0^1 g(\alpha) d\alpha &= \int_0^1 (\phi_j^\top \mu_\alpha)^2 d\alpha \\
&= \phi_j^\top \left(\int_0^1 \mu_\alpha \mu_\alpha^\top d\alpha \right) \phi_j \\
&= \phi_j^\top Z_T \phi_j = \sigma_j.
\end{aligned}$$

Next, we bound the Lipschitz constant of g .

We have

$$\|\mu_\alpha\|_2^2 = \sum_{t=0}^{T-1} \alpha^{2t} \leq T,$$

and

$$\left\| \frac{\partial}{\partial \alpha} \mu_\alpha \right\|_2^2 = \sum_{t=0}^{T-1} (t\alpha^{t-1})^2 \leq T^3.$$

Thus

$$\begin{aligned} |g'(\alpha)| &= 2|\mu_\alpha^\top \phi_j| \left| \frac{\partial}{\partial \alpha} (\mu_\alpha^\top \phi_j) \right| \\ &\leq 2\|\mu_\alpha\|_2 \|\phi_j\|_2 \left\| \frac{\partial}{\partial \alpha} \mu_\alpha \right\|_2 \\ &\leq 2\sqrt{T}\sqrt{T^3} = 2T^2. \end{aligned}$$

Hence g is L -Lipschitz with $L \leq 2T^2$.

Let $g_{\max} = \max_{\alpha \in [0,1]} g(\alpha)$. Among all nonnegative L -Lipschitz functions attaining value $g_{\max}$, the minimal area is achieved by a triangle. Because the half-width of this triangle's base is at most $g_{\max}/L \leq T/(2T^2) = 1/(2T) \leq 1/2$, the triangle cannot be truncated by both boundaries of the interval $[0, 1]$ simultaneously. Thus, the area strictly within $[0, 1]$ is at least half the total area of the unbounded triangle, yielding

$$\int_0^1 g(\alpha) d\alpha \geq \frac{g_{\max}^2}{2L}.$$

Therefore,

$$g_{\max}^2 \leq 2L\sigma_j \leq 4T^2\sigma_j.$$

Taking fourth roots,

$$\max_{\alpha} |\phi_j^\top \mu_\alpha| \leq (4T^2\sigma_j)^{1/4}.$$

Using Theorem 11.1,

$$\sigma_j \leq 4\pi^2 e^{-\frac{j}{\log T}},$$

which yields

$$|\phi_j^\top \mu_\alpha| = O\left(T^{1/2} e^{-\frac{j}{4\log T}}\right).$$

□

11.2.1 Generalization to High Dimension

Consider a linear dynamical system with matrices A, B, C . The expression $\tilde{\mathbf{u}}_t$ is now a block matrix of dimensions $T \times d_u$, and thus $\tilde{\mathbf{u}}_t^\top \mu_\alpha$ is a vector of dimension d_u .

We further assume that A is symmetric. This implies that A can be diagonalized over the set of reals as $A = U^\top D U$. Decomposing over each dimension $j \in [d_x]$ separately, and denoting $D_{jj} = \alpha_j$, we can write (11.1):

$$\begin{aligned}
 \mathbf{y}_t &= \sum_{i=1}^{t-1} C U^\top D^{i-1} U B \mathbf{u}_{t-i} \\
 &= \sum_{i=1}^{t-1} C U^\top \left(\sum_{j=1}^{d_x} \alpha_j^{i-1} e_j e_j^\top \right) U B \mathbf{u}_{t-i} \\
 &= \sum_{j=1}^{d_x} c_j b_j^\top \sum_{i=1}^{t-1} \alpha_j^{i-1} \mathbf{u}_{t-i} & c_j, b_j \leftarrow j^{\text{th}} \text{ col of } C U^\top, U B \\
 &= \sum_{j=1}^{d_x} M_j (\tilde{\mathbf{u}}_t^\top \mu_{\alpha_j}) & \text{where } M_j := c_j b_j^\top \in \mathbb{R}^{d_y \times d_u}.
 \end{aligned}$$

Thus, we can apply the same approximation techniques from the previous section, and derive an analogous conclusion to Lemma 11.3 for dimension higher than one. Crucially, the number of parameters required does not depend on the hidden dimension of the original system. The details are left as Exercise 11.2.

11.3 Online Spectral Filtering

The prediction class Π_h can be learned by online gradient descent in the online convex optimization framework, spelled out in Algorithm 7. The regret guarantees of the resulting algorithm, as given in Theorem 6.1, directly imply the following.

Theorem 11.5. *Let D be the diameter of the parameter set $\mathcal{K}$, and G an upper bound on the norm of the gradients for the loss functions f_t as in Algorithm 7. For choice of step sizes $\eta_t = \frac{D}{G\sqrt{t}}$, we have that*

$$\sum_{t=1}^T \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2 - \min_{\pi \in \Pi_h} \sum_{t=1}^T \|\mathbf{y}_t - \mathbf{y}_t^\pi\|^2 \leq 2GD\sqrt{T}.$$

Algorithm 7 Online Spectral Filtering with OGD

-
- 1: **Input:** $M_{1:h}^1$, convex constraints set $\mathcal{K} \subseteq \mathbb{R}^{h \times d_y \times d_u}$, horizon T
 - 2: Compute h top eigenvectors of Z_T , denoted $\phi_1, \dots, \phi_h$.
 - 3: **for** $t = 1$ to T **do**
 - 4: Predict $\hat{\mathbf{y}}_t = \sum_{j=1}^h M_j^t(\tilde{\mathbf{u}}_t^\top \phi_j)$ and observe true $\mathbf{y}_t$.
 - 5: Define loss $f_t(M) = \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2$.
 - 6: Update and project:

$$M^{t+1} = M^t - \eta_t \nabla f_t(M^t)$$

$$M^{t+1} = \Pi_{\mathcal{K}}(M^{t+1})$$

7: **end for**

However, the gradients of the loss can depend polynomially on T , since the feature vectors can have norms that grow with the horizon. This renders the regret guarantee of OGD uninformative in our setting.

To address this issue, we turn instead to the Vovk-Azoury-Warmuth (VAW) forecaster from Appendix A. In the spectral filtering setting, the feature vector is known before the prediction is made, so the VAW protocol is a natural fit. For simplicity of exposition, we present the scalar setting ($d_y = d_u = 1$). The resulting regret depends multiplicatively only on the label bound Y , while the dependence on the feature magnitudes appears only through a logarithmic determinant term.

Algorithm 8 Spectral Filtering with VAW

-
- 1: **Input:** Filter size h , $\lambda > 0$, horizon T .
 - 2: **Precompute:** Top h eigenvectors of Z_T , stored in $\Phi_h \in \mathbb{R}^{T \times h}$.
 - 3: **Initialize:** $A_0 = \lambda I_h$, $v_0 = 0 \in \mathbb{R}^h$.
 - 4: **for** $t = 1, 2, \dots, T$ **do**
 - 5: $\tilde{\mathbf{u}}_t = [u_{t-1}, \dots, u_1, 0, \dots, 0]^\top$.
 - 6: $h_t = \Phi_h^\top \tilde{\mathbf{u}}_t$.
 - 7: Curvature update: $A_t = A_{t-1} + h_t h_t^\top$.
 - 8: Predict $\hat{y}_t = h_t^\top A_t^{-1} v_{t-1}$ and observe y_t .
 - 9: Update target vector: $v_t = v_{t-1} + y_t h_t$.
 - 10: **end for**
-

Corollary 11.6 (VAW regret for spectral filtering). *Assume $|y_t| \leq Y$ for all t , and let $\Theta \subseteq \mathbb{R}^h$ satisfy $\|\theta\|_2 \leq B$ for all $\theta \in \Theta$. Then*

$$\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \min_{\theta \in \Theta} \sum_{t=1}^T (y_t - \theta^\top h_t)^2 \leq \lambda B^2 + Y^2 \log \det \left(I_h + \frac{1}{\lambda} \sum_{t=1}^T h_t h_t^\top \right).$$

If in addition $|u_s| \leq 1$ for all s , and setting $\lambda = 1/B^2$, gives

$$\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \min_{\theta \in \Theta} \sum_{t=1}^T (y_t - \theta^\top h_t)^2 \leq 1 + h Y^2 \log(1 + B^2 T^2).$$

Indeed, applying the VAW regret bound from Appendix A with $a_t = h_t$ and $b_t = y_t$ yields

$$\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \sum_{t=1}^T (y_t - \theta^\top h_t)^2 \leq \lambda \|\theta\|_2^2 + Y^2 \log \det \left(I_h + \frac{1}{\lambda} \sum_{t=1}^T h_t h_t^\top \right)$$

for every $\theta \in \mathbb{R}^h$. Taking θ to be the best predictor in Θ gives the corollary.

Since Φ_h has orthonormal columns, $\|\Phi_h^\top\|_2 = 1$. If $|u_s| \leq 1$, then $\|\tilde{\mathbf{u}}_t\|_2 \leq \sqrt{T}$ and hence $\|h_t\|_2 \leq \sqrt{T}$. Therefore $\|h_t h_t^\top\|_2 \leq T$, so

$$\left\| \sum_{t=1}^T h_t h_t^\top \right\|_2 \leq T^2,$$

which implies

$$\log \det \left(I_h + \frac{1}{\lambda} \sum_{t=1}^T h_t h_t^\top \right) \leq h \log \left(1 + \frac{T^2}{\lambda} \right).$$

Even if the feature magnitudes grow polynomially with T , their contribution remains only logarithmic. Thus VAW avoids the large multiplicative constants that arise in first-order methods, and the leading multiplicative dependence is only on the label magnitude Y .

11.4 Conclusion

In this chapter, we introduced Spectral Filtering as a novel approach to state estimation and prediction in linear dynamical systems, offering an alternative to classical methods like Kalman filtering and online learning of linear

predictors. Using spectral decompositions, we bypass the need for explicit system identification and provide a framework that is robust to the stability properties of the system.

We began by outlining the core challenges of traditional state estimation approaches. Kalman filtering, while optimal under Gaussian noise assumptions, requires knowledge of system matrices, which may be difficult to estimate in high-noise or adversarial settings. Online learning of linear predictors relaxed these requirements but introduced a dependence on system stability constants, which can be large and unknown. Spectral Filtering circumvents these limitations by constructing predictors in a transformed basis, allowing for efficient representation without requiring explicit system matrices.

We demonstrated the method first in the one-dimensional case, providing an intuition for how spectral predictors extract relevant dynamical structure. This was then extended to higher-dimensional settings, where spectral filtering enables a structured and computationally efficient approach to forecasting. Crucially, we showed that this approach does not rely on system stability constants, making it particularly appealing for learning in both stable and unstable dynamical systems.

From a methodological standpoint, we connected Spectral Filtering to convex optimization techniques, showing that it can be implemented via online learning in a spectral basis. This formulation allows for efficient sequential learning and prediction with provable performance guarantees.

Looking ahead, Spectral Filtering opens the door for several exciting directions. Extensions to asymmetric system matrices, as referenced in the bibliographic section, provide a pathway toward broader applicability. Moreover, integrating spectral methods with control strategies may lead to novel approaches for nonstochastic control with partial observations, which we explore in the next part of this text.

11.5 Bibliographic Remarks

Deviating from classical recovery and into the world of improper prediction, the work of [HSZ17] devised an improper learning technique using the eigenvectors of the Hankel moment matrix Z . The methods we surveyed in previous chapters for learning linear dynamical systems had statistical and computational complexity that depended on the hidden dimension. In contrast, the spectral filtering technique studied in this chapter has regret bounds and computational complexity guarantees that do not depend on the hidden dimension.

The spectral decay properties of the Hankel (Hilbert) matrices used in this chapter can be derived from the displacement-structure bounds of [BT17], and are closely connected to the classical approximation theory of Hankel operators; see, e.g., [Pel03] for a comprehensive treatment.

Connections to realization and model-reduction theory trace back to the Ho–Kalman construction [HK66] and to balanced truncation and related approximation methods [Ant05].

Spectral filtering can also be interpreted through the lens of operator approximation: it approximates the input–output map via projection onto leading modes of a compact Hankel operator. This viewpoint is closely related to Koopman/operator-theoretic approaches and data-driven spectral methods such as Dynamic Mode Decomposition [TRL⁺14].

The first extension of spectral filtering beyond symmetric settings to general (asymmetric) linear dynamical systems is due to [HLS⁺18]. More recent dimension-free advances for learning asymmetric linear dynamical systems were developed in [MH25].

The technique of spectral filtering has found applications in control and sequence prediction since its introduction. In [AHL⁺18], the authors use a spectral filtering representation to cast control of unknown linear dynamical systems as a convex program. Closed-loop control using spectral filtering techniques was recently advanced in [BBDH25].

A recent line of work makes use of spectral bases to design neural architectures for sequence prediction. This was first suggested in the context of spectral state space models [ASCH23], and was extended to hybrid models such as FlashSTU [LND⁺24]. Later extensions explored efficient implementation [ACD⁺24] and the concept of length generalization [MDA⁺24]. Structured state-space models such as S4 [GGR22] also exploit diagonalizations of linear recurrences, though derived from continuous-time parameterizations rather than Hankel moment structure.

At a conceptual level, spectral filtering replaces explicit state estimation by operator approximation: instead of reconstructing latent states, it approximates the input–output map in a near-optimal spectral basis. This perspective unifies ideas from realization theory, compact operator approximation, and modern sequence modeling.

11.6 Exercises

11.1. Generalize Lemma 11.3 to all real numbers in the range $[-1, 1]$, instead of $[0, 1]$.

11.2. Generalize Lemma 11.3 to any dimension $d_x > 1$.

11.3. The fact that Z_T is positive semidefinite is important to ensure that the eigenvalues of Z_T decay. Give an example of a family of $n \times n$ non-positive semidefinite Hankel matrices that have eigenvalues with constant absolute value.

11.4. The trace of a matrix is the sum of its eigenvalues, which measures the “total energy” of the system.

1. Using the integral definition $Z_T = \int_0^1 \mu_\alpha \mu_\alpha^\top d\alpha$, show that the (i, j) -th entry of Z_T (where indices i, j run from 0 to $T - 1$) is given by:

$$(Z_T)_{i,j} = \frac{1}{i + j + 1}.$$

2. Compute the trace of Z_T as a summation.

3. Show that $\text{Tr}(Z_T) \approx \frac{1}{2} \ln T$.

Note: This contrasts with the operator norm $\|Z_T\|_2$, which is bounded by π . The fact that the trace grows logarithmically while the norm is bounded implies that the vast majority of eigenvalues must be extremely small, reinforcing the motivation for spectral filtering.

11.5. Consider the simplest non-trivial case where the horizon is $T = 2$.

1. Compute the matrix Z_2 explicitly using the formula derived in the previous exercise.
2. Calculate the two eigenvalues of Z_2 .
3. Calculate the ratio λ_1/λ_2 . What does this ratio tell you about how much information is captured by the first spectral feature compared to the second?

11.6. In spectral filtering, we learn a parameter vector $\theta \in \mathbb{R}^h$ such that our prediction is $\hat{y}_t = \theta^\top (\Phi_h^\top \tilde{\mathbf{u}}_t)$, where $\Phi_h \in \mathbb{R}^{T \times h}$ contains the top h eigenvectors of Z_T . Standard linear predictors (FIR filters) use a weight vector $\mathbf{w} \in \mathbb{R}^T$ such that $\hat{y}_t = \mathbf{w}^\top \tilde{\mathbf{u}}_t$.

1. Find the expression for the equivalent FIR weights $\mathbf{w}_{\text{spec}}$ implied by the learned spectral parameters θ .
2. If $h = T$ (we use all eigenvectors), can we represent *any* arbitrary FIR filter $\mathbf{w}$ using spectral parameters θ ? Why or why not?

Chapter 12

Luenberger Observers

In Chapter 11 we developed spectral filtering for learning linear dynamical systems (LDS) without dependence on the hidden state dimension. However, those guarantees relied implicitly on favorable spectral structure. In particular, systems with symmetric or real spectrum are significantly easier to learn than fully asymmetric systems with complex eigenvalues near the unit circle.

In this chapter we address the learning of asymmetric linear dynamical systems. We begin by showing that without spectral restrictions, learning can be information-theoretically hard. We then introduce the Luenberger observer as a mechanism for spectrally reshaping the observer error dynamics. This leads to a regret bound governed by a new quantity: the Luenberger complexity Q_* .

12.1 Lower Bounds for Learning Asymmetric LDS

We first show that without spectral restrictions, learning an LDS may require a number of samples proportional to the hidden dimension.

Theorem 12.1. *For any online learning algorithm $\mathcal{A}$, there exists a linear dynamical system with hidden state dimension d_h such that the expected cumulative squared loss of $\mathcal{A}$ over $T \geq d_h$ rounds is at least $\Omega(d_h)$.*

Proof. We construct an autonomous LDS for which prediction is equivalent to predicting independent coin flips.

Consider the system:

$$\begin{aligned}x_{t+1} &= Ax_t, \\ y_t &= Cx_t,\end{aligned}$$

where A is the $d_h \times d_h$ cyclic permutation matrix

$$A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

and $C = (1, 0, \dots, 0)$.

Let the initial state $x_0 \in \{-1, 1\}^{d_h}$ be drawn uniformly at random, with independent Rademacher entries.

Then $x_t = A^t x_0$, and for $t = 0, \dots, d_h - 1$,

$$y_t = (x_0)_{t+1}.$$

Thus each observation reveals a fresh independent coordinate of x_0 .

Fix any realization of the learner's internal randomness. Conditional on this realization, the prediction at time t has the form

$$\hat{y}_t = f_t(y_0, \dots, y_{t-1}),$$

which depends only on $(x_0)_1, \dots, (x_0)_t$.

But $y_t = (x_0)_{t+1}$ is independent of these values, hence

$$\mathbb{E}[y_t \mid y_0, \dots, y_{t-1}] = 0.$$

Therefore,

$$\begin{aligned} \mathbb{E}[(y_t - \hat{y}_t)^2] &= \mathbb{E}[y_t^2 - 2y_t\hat{y}_t + \hat{y}_t^2] \\ &= 1 + \mathbb{E}[\hat{y}_t^2] \\ &\geq 1. \end{aligned}$$

Summing over $t = 0, \dots, d_h - 1$ yields expected cumulative loss at least d_h . $\square$

This lower bound shows that without restricting spectral structure, learning asymmetric systems may require a number of samples proportional to the hidden dimension.

We now show that this hardness can be overcome when the system admits an observer representation that reshapes the error dynamics.

12.2 Pole Placement

In this section we prove a fundamental and elegant result from classical control theory: the eigenstructure of the observer error dynamics can be modified completely under the observability assumption.

The key structural condition that allows this radical shift in eigenstructure is *observability*, as defined earlier in the book. Recall that (A, C) is observable if the observability matrix

$$\mathcal{O}_{d_h} = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{d_h-1} \end{pmatrix}$$

has full column rank. Equivalently, the only state x satisfying $CA^k x = 0$ for all $k \geq 0$ is $x = 0$. Observability ensures that the hidden state can be completely reconstructed from the outputs.

The following result shows that this same condition allows for the arbitrary assignment of the spectrum of the observer error dynamics.

Theorem 12.2 (Observer Pole Placement). *Let (A, C) be observable. Then for any self-conjugate set of d_h complex numbers Λ , there exists an observer gain matrix $L \in \mathbb{R}^{d_h \times d_y}$ such that the spectrum (the set of eigenvalues) of $A - LC$ is exactly Λ .*

Proof. For clarity we provide a constructive, self-contained proof for the single-output case ($d_y = 1$), where C is a $1 \times d_h$ row vector and L is a $d_h \times 1$ column vector.

Let the characteristic polynomial of the unobserved dynamics A be

$$p(z) = \det(zI - A) = z^{d_h} + a_{d_h-1}z^{d_h-1} + \cdots + a_1z + a_0.$$

Because the pair (A, C) is observable, there exists an invertible change-of-basis matrix T such that in the new coordinates $x = T\bar{x}$, the system matrices $\bar{A} = T^{-1}AT$ and $\bar{C} = CT$ are in *Observable Canonical Form*:

$$\bar{A} = \begin{pmatrix} 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & \cdots & 0 & -a_1 \\ 0 & 1 & \cdots & 0 & -a_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -a_{d_h-1} \end{pmatrix}, \quad \bar{C} = (0 \ 0 \ \cdots \ 0 \ 1).$$

Let the desired target spectrum Λ be defined by the roots of a monic polynomial:

$$q(z) = z^{d_h} + \tilde{a}_{d_h-1}z^{d_h-1} + \cdots + \tilde{a}_1z + \tilde{a}_0.$$

We wish to find an observer gain in the transformed coordinates, $\bar{L} = (l_0, l_1, \dots, l_{d_h-1})^\top$, such that $\bar{A} - \bar{L}\bar{C}$ has characteristic polynomial $q(z)$.

Observe the structure of the outer product $\bar{L}\bar{C}$: it is a $d_h \times d_h$ matrix of all zeros, except for its last column which equals exactly $\bar{L}$. Therefore, subtracting it from $\bar{A}$ simply modifies the last column:

$$\bar{A} - \bar{L}\bar{C} = \begin{pmatrix} 0 & 0 & \cdots & 0 & -(a_0 + l_0) \\ 1 & 0 & \cdots & 0 & -(a_1 + l_1) \\ 0 & 1 & \cdots & 0 & -(a_2 + l_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -(a_{d_h-1} + l_{d_h-1}) \end{pmatrix}.$$

Because this resulting matrix is also in canonical form, its characteristic polynomial can be read directly from its last column:

$$\det(zI - (\bar{A} - \bar{L}\bar{C})) = z^{d_h} + (a_{d_h-1} + l_{d_h-1})z^{d_h-1} + \cdots + (a_0 + l_0).$$

To assign the spectrum Λ , we simply match the coefficients to $q(z)$ by choosing $l_i = \tilde{a}_i - a_i$ for all $i = 0, \dots, d_h - 1$.

Finally, we transform the gain back to the original coordinates by setting $L = T\bar{L}$. The closed-loop matrix is $A - LC = T(\bar{A} - \bar{L}\bar{C})T^{-1}$, which is similar to $\bar{A} - \bar{L}\bar{C}$ and therefore possesses the exact assigned spectrum Λ . $\square$

Constructing the observable canonical form. In the single-output case ($d_y = 1$), the observability matrix

$$\mathcal{O}_{d_h}(A, C) = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{d_h-1} \end{pmatrix}$$

is a $d_h \times d_h$ matrix. Since (A, C) is observable, it has full rank and is therefore invertible. Likewise, the observability matrix of the observable companion pair $(A_{\text{obs}}, C_{\text{obs}})$ is invertible. Because under the coordinate change $x = T\bar{x}$ we have

$$\mathcal{O}_{d_h}(\bar{A}, \bar{C}) = \mathcal{O}_{d_h}(A, C)T,$$

we may compute the similarity transform from

$$\mathcal{O}_{d_h}(A, C)T = \mathcal{O}_{d_h}(A_{\text{obs}}, C_{\text{obs}}), \quad \text{so} \quad T = \mathcal{O}_{d_h}(A, C)^{-1} \mathcal{O}_{d_h}(A_{\text{obs}}, C_{\text{obs}}).$$

We proceed to show how pole placement can be used to create the Luenberger observer system: a powerful technique in learning of dynamical systems.

12.3 The Luenberger Observer

Consider the linear dynamical system

$$x_{t+1} = Ax_t + Bu_t, \quad y_t = Cx_t.$$

Definition 12.3 (Luenberger Observer). *Given a gain matrix $L \in \mathbb{R}^{d_h \times d_y}$, the Luenberger observer is the dynamical system*

$$\tilde{x}_{t+1} = A\tilde{x}_t + Bu_t + L(y_t - \tilde{y}_t), \quad \tilde{y}_t = C\tilde{x}_t.$$

Define the estimation error

$$e_t := x_t - \tilde{x}_t.$$

Subtracting the observer dynamics from the true dynamics yields

$$\begin{aligned} e_{t+1} &= x_{t+1} - \tilde{x}_{t+1} \\ &= (Ax_t + Bu_t) - (A\tilde{x}_t + Bu_t + L(y_t - \tilde{y}_t)) \\ &= (A - LC)e_t. \end{aligned}$$

Thus the observer error evolves autonomously under the matrix $A - LC$, entirely independent of the control inputs u_t . By Theorem 12.2, if (A, C) is observable then L can be chosen so that the spectrum of $A - LC$ matches any desired target spectrum. We define the resulting closed-loop matrix as $\tilde{A} := A - LC$.

12.3.1 Luenberger Complexity

Although pole placement guarantees that we may assign the eigenvalues of $\tilde{A}$ arbitrarily, the transient behavior of the observer depends not only on the spectrum, but also on the conditioning of the eigenbasis.

We henceforth suppose that the chosen closed-loop matrix $\tilde{A} = A - LC$ is diagonalizable. For any diagonalization $\tilde{A} = HDH^{-1}$, the factor $\|H\| \|H^{-1}\|$ measures the conditioning of the eigenbasis.

Definition 12.4 (Luenberger Complexity). *For a strictly stable, diagonalizable matrix $\tilde{A}$, let $\gamma = 1 - \rho(\tilde{A}) > 0$ be its stability margin. Define*

$$\kappa_{\text{diag}}(\tilde{A}) = \min\{\|H\| \|H^{-1}\| : \tilde{A} = HDH^{-1}, D \text{ diagonal}\},$$

and its Luenberger complexity by

$$Q(\tilde{A}) := \frac{1}{\gamma} \max(1, \log \kappa_{\text{diag}}(\tilde{A})).$$

Thus $Q(\tilde{A})$ measures the transient amplification and time scale required before contraction dominates. By definition, $Q(\tilde{A}) \geq 1/\gamma$.

Theorem 12.5 (Error Contraction and Burn-In). *Let (A, C) be observable, choose L , and let $\tilde{A} = A - LC$. Assume $\tilde{A}$ is strictly stable and diagonalizable with stability margin $\gamma = 1 - \rho(\tilde{A}) > 0$. Then*

$$\|e_t\| \leq \kappa_{\text{diag}}(\tilde{A}) (1 - \gamma)^t \|e_0\|.$$

Consequently, for any $\varepsilon > 0$, it suffices that

$$t \geq \frac{1}{\gamma} \left(\log \kappa_{\text{diag}}(\tilde{A}) + \log(\|e_0\|/\varepsilon) \right)$$

to guarantee $\|e_t\| \leq \varepsilon$. In particular, for a fixed initial error and desired accuracy, the burn-in time satisfies

$$t_{\text{burn}} = O(Q(\tilde{A})).$$

Proof. Let $\tilde{A} = HDH^{-1}$ be a diagonalization attaining the minimum in the definition of $\kappa_{\text{diag}}(\tilde{A})$. Since $e_{t+1} = \tilde{A}e_t$, we have $e_t = \tilde{A}^t e_0$. Using the diagonalization $\tilde{A} = HDH^{-1}$,

$$\tilde{A}^t = HD^t H^{-1}.$$

Thus

$$\|\tilde{A}^t\| \leq \|H\| \|H^{-1}\| \|D^t\| \leq \kappa_{\text{diag}}(\tilde{A}) (1 - \gamma)^t.$$

Multiplying by $\|e_0\|$ yields the first inequality. The burn-in bound follows by solving

$$\kappa_{\text{diag}}(\tilde{A}) (1 - \gamma)^t \|e_0\| \leq \varepsilon,$$

and using the standard inequality $\log(1 - \gamma) \leq -\gamma$. □

Interpretation. Pole placement ensures existence of a stable observer. However, the effective convergence speed depends on both the spectral gap γ and the conditioning of the eigenbasis (which is constrained because we can only assign the eigenvalues, not the exact matrix $\tilde{A}$). The logarithmic Luenberger complexity $Q(\tilde{A})$ captures precisely the scale that governs this burn-in time.

12.3.2 One Dimensional Example

Consider the simple one dimensional (and thus fully observable) linear dynamical system

$$x_{t+1} = 2x_t, \quad y_t = x_t.$$

Define

$$\tilde{x}_{t+1} = 2\tilde{x}_t + L(x_t - \tilde{x}_t).$$

Then

$$e_{t+1} = (2 - L)e_t.$$

Choosing $L = 2.5$ yields

$$e_{t+1} = -0.5e_t,$$

so the error contracts exponentially even though the system diverges.

12.4 Reduction to Stable Learning

We now exploit the reshaping principle as a general reduction for online learning. Section 12.3 showed that if (A, C) is observable, we may choose an observer gain L so that the error dynamics evolve under a strictly stable closed-loop matrix $\tilde{A} = A - LC$. The effective transient of this observer is governed by the Luenberger complexity $Q(\tilde{A})$.

Crucially, this transforms the problem of predicting an unstable, partially-observed system with infinite memory into the problem of predicting the outputs of a strictly stable, well-behaved system.

12.4.1 A Learnable Spectral Region

To guarantee that the observer forgets its initial estimation error e_0 , we must ensure that the closed-loop matrix $\tilde{A}$ is strictly stable. Fix a desired stability margin $\gamma \in (0, 1)$ and define the target spectral region simply as the strictly stable complex disk:

$$\Sigma_\gamma := \mathbb{D}_{1-\gamma} = \{z \in \mathbb{C} : |z| \leq 1 - \gamma\}.$$

We enforce the margin γ entirely to guarantee rapid error dissipation (burn-in). Marginally stable modes (e.g., $|\lambda| = 1$) are strictly excluded. Because the Luenberger observer is a closed-loop system, an eigenvalue of 1 would cause the unforced initial estimation error $e_t = \tilde{A}^t e_0$ to persist indefinitely as a constant offset. Because online learning algorithms build predictive features solely from forced inputs and outputs, they cannot accurately predict this unforced hidden offset, resulting in linear regret.

All eigenvalues of the original plant A — in particular unstable modes or complex eigenvalues near the unit circle — constitute the difficult modes; we aim to move them strictly into the stable region Σ_γ via the observer.

12.4.2 The Luenberger Program

Pole placement guarantees feasibility under observability, but not necessarily good conditioning. Even if there exists an L such that the spectrum satisfies $\sigma(A - LC) \subseteq \Sigma_\gamma$, the resulting observer may have a poorly conditioned eigenbasis, leading to a large Luenberger complexity.

We therefore define an optimization program to find the best-conditioned observer among all feasible pole placements for an arbitrary stable target set Σ .

Definition 12.6 (Luenberger Program). *Fix an arbitrary strictly stable spectral constraint set $\Sigma \subseteq \mathbb{D}_{1-\varepsilon}$ for some $\varepsilon > 0$. Define the optimal Luenberger complexity as:*

$$Q_\star(A, C; \Sigma) := \min_L Q(A - LC) \quad \text{s.t.} \quad \sigma(A - LC) \subseteq \Sigma,$$

with the convention $Q_\star(A, C; \Sigma) = +\infty$ if infeasible. For our specific stable disk $\Sigma = \Sigma_\gamma$, we write $Q_\star^\gamma$.

12.4.3 Regret Reduction

Assume $Q_\star(A, C; \Sigma_\gamma) < \infty$ and let $L^\star$ attain the minimum in the Luenberger program. Define the optimal observer matrix

$$\tilde{A} := A - L^\star C, \quad Q_\star := Q(\tilde{A}).$$

By recursively expanding the observer state, we can express the true system output y_t entirely as a function of the past observations y_{t-i} and

controls u_{t-i} , plus an unforced error term:

$$y_t = C\tilde{A}^t\tilde{x}_0 + \underbrace{\sum_{i=1}^t C\tilde{A}^{i-1}(Bu_{t-i} + L^*y_{t-i})}_{=: \bar{y}_t} + C\tilde{A}^t e_0.$$

The sequence $\bar{y}_t$ represents a strictly stable mapping from past data to the current output. Because $\tilde{A}$ is strictly stable, the sequence of coefficient matrices $C\tilde{A}^{i-1}B$ and $C\tilde{A}^{i-1}L^*$ decay to zero exponentially fast.

This provides a powerful, general reduction for online control: we have reduced the problem to learning a stable transfer function.

Because the coefficients decay exponentially fast, the mapping can be effectively approximated using a finite memory length $m = O(Q_* \log T)$. We may now employ *any* suitable online sequence prediction algorithm to learn this mapping. For example:

- **Linear Regression (FIR Filters):** We can parameterize the predictor as a standard finite impulse response filter of length m and learn the coefficients using Online Gradient Descent.
- **Spectral Filtering & ONS:** We can project the observations onto the highly efficient Hankel basis (which requires only $O(\log^2 T)$ features for real modes) and use the Online Newton Step to achieve logarithmic regret.

Regardless of the chosen learning algorithm, the total regret against the optimal observer is bounded by the algorithm's inherent regret for stable sequence prediction, plus an initial, bounded penalty from the Luenberger burn-in period.

The true observation y_t differs from the idealized stable prediction $\bar{y}_t$ only by the unforced observer error $C\tilde{A}^t e_0$. By Theorem 12.5, the total squared error introduced by the observer's transient burn-in over all time is bounded by a constant independent of T :

$$\begin{aligned} E_{\text{burn-in}} &:= \sum_{t=1}^T \|y_t - \bar{y}_t\|^2 \leq \|C\|^2 \kappa_{\text{diag}}(\tilde{A})^2 \|e_0\|^2 \sum_{t=1}^{\infty} (1 - \gamma)^{2t} \\ &\leq \frac{\|C\|^2 \kappa_{\text{diag}}(\tilde{A})^2 \|e_0\|^2}{2\gamma}. \end{aligned}$$

Notice that this burn-in constant is governed exactly by $\kappa_{\text{diag}}(\tilde{A})$ and $1/\gamma$, which are the quantities minimized by the Luenberger complexity Q_* .

Let $\mathcal{F}_\gamma$ denote any class of stable linear predictors with margin γ that contains the stable surrogate sequence $\bar{y}_t$ defined above.

Theorem 12.7 (Regret Reduction). *Let $\mathcal{A}$ be an online learning algorithm that guarantees $\text{Regret}_{\mathcal{A}}(T)$ against the class $\mathcal{F}_\gamma$. If we apply $\mathcal{A}$ to the sequence of observations from the partially observed system (A, C) , then*

$$\sum_{t=1}^T \|y_t - \hat{y}_t\|^2 \leq \inf_{f \in \mathcal{F}_\gamma} \sum_{t=1}^T \|y_t - f_t\|^2 + \text{Regret}_{\mathcal{A}}(T) \leq \text{Regret}_{\mathcal{A}}(T) + E_{\text{burn-in}}(L^*).$$

This reduction completely separates the *control* problem (using observability to reshape the error dynamics and minimize Q_*) from the *learning* problem (running an algorithm on a stable system). The regret against the true underlying (potentially unstable and asymmetric) system is therefore governed by the regret of learning a stable system, plus a constant burn-in penalty.

12.5 Summary

This chapter shows that partial observation fundamentally changes the learning problem for linear dynamical systems. Without additional structure, one cannot hope for dimension-free guarantees in general: the lower bound demonstrates that a learner may be forced to uncover the hidden state one coordinate at a time, incurring loss of order d_h before the system becomes predictable. Thus the hidden dimension is a genuine barrier if we work directly with the original dynamics.

The Luenberger observer provides a way around this obstacle. When (A, C) is observable, we may introduce an output-injection matrix L and study the estimation error dynamics

$$e_{t+1} = (A - LC)e_t.$$

Pole placement shows that the spectrum of this closed-loop matrix can be moved into any prescribed stable region. This does not by itself solve the learning problem, because fast asymptotic decay is not enough: one must also control transient growth. That is the role of the Luenberger complexity $Q(\tilde{A})$, and of the optimized quantity Q_* , which measure how quickly the observer forgets its initial error and how badly the non-normal geometry can amplify that error before contraction dominates.

Once a well-conditioned observer is chosen, the original partially observed system is converted into a stable input-output prediction problem, up to a

burn-in term coming from the decaying initial estimation error. The chapter therefore separates into two clean pieces. Control theory supplies the observer that reshapes the error dynamics into a stable form, and online learning supplies an algorithm for the resulting stable sequence-prediction task. In this way, the effective difficulty of learning is governed not directly by the hidden dimension d_h , but by the best attainable observer geometry encoded by $Q_\star$.

12.6 Bibliographic Remarks

The fundamental hardness of learning linear dynamical systems without spectral assumptions, as discussed in the beginning of this chapter, contrasts with the symmetric or stable cases. While [HMR18] showed that gradient descent efficiently learns stable systems, the lower bounds for general asymmetric systems typically rely on constructions where the system dynamics obscure the initial state for long periods, similar to the hardness results in [SMT⁺18]. The information theoretic lower bound described here is due to [HM26].

The theoretical foundation for reconstructing internal states from outputs lies in the work of [Lue64], who introduced the concept of the state observer. The specific architecture discussed in this chapter is widely known as the Luenberger Observer. [Lue71] provides a comprehensive introduction to this theory. The powerful result that observability allows for arbitrary assignment of the closed-loop spectrum (Theorem 12.2) is known as the Pole Placement Theorem, originally established by [Won67].

While classical control theory focuses on asymptotic stability, this chapter emphasizes the transient behavior of the error dynamics, which is crucial for finite-time learning guarantees. The study of transient growth in stable but non-normal matrices (where eigenvectors are ill-conditioned) is famously treated in the text on pseudospectra by [TE05]. The concept of *Luenberger Complexity* ($Q_\star$) serves as a quantitative measure of this transient behavior specifically optimized for the learning setting. It was introduced in [DBH25].

The reduction framework presented in this chapter, which utilizes the Luenberger observer to transform an unstable or asymmetric learning problem into a stable prediction task, is based on the work of [DBH25]. This work showed that dimension-free regret bounds are achievable for general linear systems provided one can find a well-conditioned observer, shifting the complexity burden from the state dimension d_h to the spectral conditioning of the dynamics.

12.7 Exercises

12.1. This exercise illustrates the "peaking phenomenon," where placing eigenvalues far into the stable region can cause the error norm to grow transiently before decaying (i.e., high Luenberger complexity). Consider the two-dimensional system with $A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ and $C = [1 \ 0]$.

1. Verify that the system is observable.
2. Find the observer gain vector $L = [l_1, l_2]^\top$ such that the closed-loop matrix $\tilde{A} = A - LC$ has both eigenvalues equal to some $\lambda \in [0, 1)$.
3. Show that as we demand faster convergence (i.e., as $\lambda \rightarrow 0$), the norm of the gain $\|L\|$ tends to infinity.
4. Calculate the matrix $\tilde{A}$ explicitly for $\lambda = 0$. Compute $\|\tilde{A}e_0\|$ for an initial error $e_0 = [0, 1]^\top$. What happens to the error magnitude in the first step?

12.2. A *deadbeat observer* is one where the closed-loop poles are all placed at the origin (0).

1. Let $\tilde{A} = A - LC$. Prove that if all eigenvalues of $\tilde{A}$ are 0, then $\tilde{A}$ is nilpotent, meaning $\tilde{A}^{d_h} = 0$.
2. Conclude that for a deadbeat observer, the estimation error e_t becomes exactly zero for all $t \geq d_h$, regardless of the initial error.
3. Why might we not always choose a deadbeat observer in practice, particularly in the presence of observation noise $\mathbf{v}_t$? (Hint: Consider the magnitude of L derived in Exercise 12.1).

12.3. In Theorem 12.1, we used a cyclic permutation matrix to prove a lower bound on learning. Let A be the $d_h \times d_h$ cyclic permutation matrix where $A_{i,i+1} = 1$ for $1 \leq i < d_h$, $A_{d_h,1} = 1$, and 0 elsewhere. Let $C = [1, 0, \dots, 0]$.

1. Write down the observability matrix $\mathcal{O}_{d_h}$ for this pair (A, C) .
2. Prove that this system is fully observable.
3. Since it is observable, Theorem 12.2 implies we can stabilize it. Find a gain vector L such that $A - LC$ is nilpotent.

12.4. Prove that the set of stable matrices is not convex. specifically, give two matrices $A_1, A_2 \in \mathbb{R}^{2 \times 2}$ such that $\rho(A_1) < 1$ and $\rho(A_2) < 1$, but $\rho(\frac{1}{2}A_1 + \frac{1}{2}A_2) > 1$. *Note: This geometric fact implies that the Luenberger Program (Definition 12.6), which optimizes over stable matrices, involves a non-convex constraint set.*

Chapter 13

Learning in Nonlinear Dynamical Systems

In the previous chapters, we developed a comprehensive theory for learning and predicting in linear dynamical systems. We moved from symmetric systems in Chapter 11 to general, asymmetric, and potentially unstable linear systems in Chapter 12 via the Luenberger observer framework.

However, the physical world is rarely strictly linear. In this chapter, we address the fundamental problem of learning *nonlinear* dynamical systems. Formally, we consider a system of the form:

$$\begin{aligned}\mathbf{x}_{t+1} &= f(\mathbf{x}_t) \\ \mathbf{y}_t &= h(\mathbf{x}_t)\end{aligned}\tag{13.1}$$

where $f : \mathcal{X} \rightarrow \mathcal{X}$ is the nonlinear transition function and $h : \mathcal{X} \rightarrow \mathbb{R}^{d_y}$ is the observation function. The state $\mathbf{x}_t$ is hidden, and the functions f and h are unknown. For clarity, we treat the autonomous case in this chapter; the controlled extension $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t)$ is significantly more complex.

Historically, learning such systems involved *system identification*: attempting to approximate the function f using neural networks or kernels. In this chapter, we take a different approach rooted in improper learning. Rather than trying to recover the nonlinearity explicitly, we compete with a linear surrogate that predicts well.

More precisely, we will show two things. First, every bounded nonlinear system with non-expansive dynamics admits a finite-dimensional approximate linear representation over any fixed horizon via discretization. Second, if such a lifted linear approximation admits a stable, well-conditioned observer, then the learning guarantees from Chapter 12 allow us to compete

with it. Thus, nonlinear prediction can be reduced to linear prediction, provided the corresponding lifted system has moderate observer complexity.

13.1 Global Linear Approximation via Discretization

The core mathematical insight enabling linear learning of nonlinear dynamics is that nonlinear dynamics in a finite-dimensional space can be lifted to linear dynamics in a higher, potentially infinite-dimensional space. This is often associated with the *Koopman operator*, which we discuss in Section 13.2.

We begin with a concrete finite-horizon construction based on discretization.

13.1.1 The ε -Net Construction

We make the following standard assumptions about the regularity of the system:

1. **Boundedness:** The state space $\mathcal{X}$ is contained in a ball of radius R .
2. **Lipschitz continuity:** The dynamics f and observation map h are 1-Lipschitz. That is,

$$\|f(x) - f(z)\| \leq \|x - z\|, \quad \|h(x) - h(z)\| \leq \|x - z\|$$

for all $x, z \in \mathcal{X}$.

Let $\mathcal{S}$ be an ε -net of the state space $\mathcal{X}$. This is a finite set of points

$$\mathcal{S} = \{s^{(1)}, \dots, s^{(N)}\}$$

such that for every $x \in \mathcal{X}$, there exists some $s \in \mathcal{S}$ with $\|x - s\| \leq \varepsilon$. Its cardinality satisfies

$$N = O\left((R/\varepsilon)^{d_x}\right).$$

We define a lifted state space of dimension N , where the lifted state $\mathbf{z}_t \in \mathbb{R}^N$ is a one-hot vector encoding the current approximate grid location. Let $\pi(x)$ denote a nearest grid point to x in $\mathcal{S}$, with ties broken arbitrarily:

$$\pi(x) \in \arg \min_{s \in \mathcal{S}} \|x - s\|.$$

We now define a linear transition matrix $A' \in \mathbb{R}^{N \times N}$ and observation matrix $C' \in \mathbb{R}^{d_y \times N}$ as follows:

1. **Dynamics (A'):** If the system is at grid point $s^{(j)}$, the next state is approximated by the grid point nearest to $f(s^{(j)})$. Thus

$$A'_{ij} = 1 \quad \text{if} \quad s^{(i)} = \pi(f(s^{(j)})), \quad A'_{ij} = 0 \quad \text{otherwise.}$$

2. **Observations (C'):** The j -th column of C' is the observation vector at the grid point $s^{(j)}$:

$$C'_{:,j} = h(s^{(j)}).$$

This defines a linear system:

$$\mathbf{z}_{t+1} = A' \mathbf{z}_t, \quad \mathbf{y}'_t = C' \mathbf{z}_t.$$

If $s^{(j)} = \pi(\mathbf{x}_0)$ and we initialize $\mathbf{z}_0 = \mathbf{e}_j$, then $\mathbf{z}_t$ remains a one-hot vector for all t .

13.1.2 Approximation Guarantee

We now prove that this high-dimensional linear system remains close to the true nonlinear trajectory over a finite horizon.

Theorem 13.1 (Finite-Horizon Linear Approximation via Discretization). *Let $\mathbf{y}_0, \dots, \mathbf{y}_{T-1}$ be generated by the nonlinear system (13.1) under the boundedness and 1-Lipschitz assumptions above. Let A', C' be the matrices constructed from an ε -net $\mathcal{S}$, and let $\mathbf{y}'_t$ be the output of the lifted linear system initialized at the one-hot vector corresponding to $\pi(\mathbf{x}_0)$.*

Then for every $t = 0, 1, \dots, T-1$,

$$\|\mathbf{y}_t - \mathbf{y}'_t\| \leq (t+1)\varepsilon.$$

Consequently,

$$\sum_{t=0}^{T-1} \|\mathbf{y}_t - \mathbf{y}'_t\|^2 \leq \varepsilon^2 \sum_{t=0}^{T-1} (t+1)^2 = \varepsilon^2 \frac{T(T+1)(2T+1)}{6} \leq T^3 \varepsilon^2.$$

Proof. Let $\mathbf{x}_0, \dots, \mathbf{x}_T$ be the true trajectory of the nonlinear system. Let s_t be the grid point occupied by the lifted linear system at time t . Then

$$s_0 = \pi(\mathbf{x}_0), \quad s_{t+1} = \pi(f(s_t)).$$

Define the tracking error

$$\xi_t := \|s_t - \mathbf{x}_t\|.$$

We analyze its evolution:

$$\begin{aligned}\xi_{t+1} &= \|s_{t+1} - \mathbf{x}_{t+1}\| \\ &= \|\pi(f(s_t)) - f(\mathbf{x}_t)\|.\end{aligned}$$

By the triangle inequality,

$$\xi_{t+1} \leq \|\pi(f(s_t)) - f(s_t)\| + \|f(s_t) - f(\mathbf{x}_t)\|.$$

The first term is at most ε by the definition of the ε -net and nearest-neighbor projection. The second term is at most $\|s_t - \mathbf{x}_t\| = \xi_t$ by the 1-Lipschitz property of f . Hence

$$\xi_{t+1} \leq \xi_t + \varepsilon.$$

Since

$$\xi_0 = \|\pi(\mathbf{x}_0) - \mathbf{x}_0\| \leq \varepsilon,$$

an induction gives

$$\xi_t \leq (t+1)\varepsilon \quad \text{for all } t \geq 0.$$

Now $\mathbf{y}_t = h(\mathbf{x}_t)$ and $\mathbf{y}'_t = h(s_t)$, so by the 1-Lipschitz property of h ,

$$\|\mathbf{y}_t - \mathbf{y}'_t\| = \|h(\mathbf{x}_t) - h(s_t)\| \leq \|\mathbf{x}_t - s_t\| = \xi_t \leq (t+1)\varepsilon.$$

This proves the pointwise bound. Summing the squared bound over $t = 0, \dots, T-1$ yields

$$\sum_{t=0}^{T-1} \|\mathbf{y}_t - \mathbf{y}'_t\|^2 \leq \varepsilon^2 \sum_{t=0}^{T-1} (t+1)^2 = \varepsilon^2 \frac{T(T+1)(2T+1)}{6} \leq T^3 \varepsilon^2.$$

□

Thus, by choosing $\varepsilon \approx T^{-3/2}$, the cumulative squared approximation error is $O(1)$ over the horizon T . The price is that the lifted dimension

$$N = O\left((R/\varepsilon)^{d_x}\right)$$

may be exponentially large in the original state dimension d_x .

13.2 The Koopman Operator

The discretization argument above gives a finite-dimensional linear approximation over a fixed horizon. The Koopman perspective explains why linear structure is present more fundamentally.

The **Koopman operator** $\mathcal{K}$ is an infinite-dimensional linear operator acting on scalar observable functions $g : \mathcal{X} \rightarrow \mathbb{R}$ by

$$(\mathcal{K}g)(x) = g(f(x)).$$

Even if f is nonlinear, the operator $\mathcal{K}$ is linear.

Thus every nonlinear dynamical system admits an exact linear representation at the level of observables, but in general this representation lives in an infinite-dimensional function space. The discretization in Section 13.1 should be viewed as a concrete finite-state approximation to this principle: by tracking which cell of an ε -net the system occupies, we obtain a finite-dimensional lifted linear system.

For the purposes of prediction, our perspective is pragmatic. We do not need to recover the full infinite-dimensional Koopman operator. We only need to know that a finite-dimensional linear approximation exists and that this approximation can be learned competitively by the online algorithms developed earlier.

13.3 Improper Learning of Nonlinear Dynamics

We can now combine the existence result of Theorem 13.1 with the learning algorithms from Chapter 12.

The key point is that the lifted linear system constructed above can have enormous hidden dimension

$$N = O\left((R/\varepsilon)^{d_x}\right),$$

yet the regret bounds from Chapter 12 do not depend directly on that hidden dimension. This makes improper learning possible: we never manipulate the lifted state $\mathbf{z}_t$ explicitly, but instead compete with the predictions of the lifted linear surrogate on the observed sequence $\mathbf{y}_t$.

Dimension-independence alone, however, is not enough. To apply the Luenberger reduction from Chapter 12, the lifted system must admit a stable, well-conditioned observer. This leads to the following complexity measure.

Definition 13.2 (Nonlinear Luenberger Complexity). *Fix $\varepsilon > 0$ and a stability margin $\gamma \in (0, 1)$. Define*

$$Q_{\star}^{\gamma}(f, h; \varepsilon) := \inf_{\substack{\mathcal{S} \text{ an } \varepsilon\text{-net of } \mathcal{X} \\ \pi \text{ a nearest-neighbor map}}} \inf_{\substack{L \\ \begin{array}{l} A' - LC' \text{ diagonalizable} \\ \sigma(A' - LC') \subseteq \mathbb{D}_{1-\gamma} \end{array}}} Q(A' - LC'),$$

where (A', C') are the lifted matrices induced by $(\mathcal{S}, \pi)$ as in Section 13.1. If no feasible choice exists, we set

$$Q_{\star}^{\gamma}(f, h; \varepsilon) = +\infty.$$

This quantity asks whether some ε -resolution linearization of the nonlinear system can be stabilized by a Luenberger observer with moderate transient behavior. Feasibility is not automatic: unlike the setting of Chapter 12, the discretized pair (A', C') need not be observable, and the nonlinear Luenberger program may therefore have value $+\infty$.

We now combine the discretization error with the linear regret bound.

Theorem 13.3 (Improper Learning Guarantee). *Fix $\varepsilon > 0$ and $\gamma \in (0, 1)$. Let (A', C') be any lifted linear system obtained from an ε -net construction in Section 13.1, and suppose there exists a gain matrix L such that $A' - LC'$ is diagonalizable and*

$$\sigma(A' - LC') \subseteq \mathbb{D}_{1-\gamma}.$$

Let $\bar{\mathbf{y}}_t$ denote the stable surrogate sequence obtained by applying the Luenberger reduction of Chapter 12 to the lifted linear system (A', C') , and let $E_{\text{burn-in}}(A', C', L)$ denote the corresponding burn-in term.

If $\mathcal{A}$ is any online learning algorithm with regret $\text{Regret}_{\mathcal{A}}(T)$ against the class of stable linear predictors with margin γ , then its predictions $\hat{\mathbf{y}}_t$ on the nonlinear sequence satisfy

$$\sum_{t=0}^{T-1} \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2 \leq \text{Regret}_{\mathcal{A}}(T) + 2T^3\varepsilon^2 + 2E_{\text{burn-in}}(A', C', L).$$

Proof. Let $\mathbf{y}'_t$ denote the output of the lifted linear system (A', C') constructed from the ε -net. By Theorem 13.1,

$$\sum_{t=0}^{T-1} \|\mathbf{y}_t - \mathbf{y}'_t\|^2 \leq T^3\varepsilon^2.$$

Next, applying the reduction from Chapter 12 to the lifted linear system (A', C') yields a stable surrogate sequence $\bar{\mathbf{y}}_t$ such that

$$\sum_{t=0}^{T-1} \|\mathbf{y}'_t - \bar{\mathbf{y}}_t\|^2 \leq E_{\text{burn-in}}(A', C', L).$$

Since $\bar{\mathbf{y}}_t$ belongs to the comparator class of stable linear predictors with margin γ , the regret guarantee of $\mathcal{A}$ gives

$$\sum_{t=0}^{T-1} \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2 \leq \sum_{t=0}^{T-1} \|\mathbf{y}_t - \bar{\mathbf{y}}_t\|^2 + \text{Regret}_{\mathcal{A}}(T).$$

Finally, using

$$\|a + b\|^2 \leq 2\|a\|^2 + 2\|b\|^2,$$

with

$$a = \mathbf{y}_t - \mathbf{y}'_t, \quad b = \mathbf{y}'_t - \bar{\mathbf{y}}_t,$$

we obtain

$$\begin{aligned} \sum_{t=0}^{T-1} \|\mathbf{y}_t - \bar{\mathbf{y}}_t\|^2 &\leq 2 \sum_{t=0}^{T-1} \|\mathbf{y}_t - \mathbf{y}'_t\|^2 + 2 \sum_{t=0}^{T-1} \|\mathbf{y}'_t - \bar{\mathbf{y}}_t\|^2 \\ &\leq 2T^3\varepsilon^2 + 2E_{\text{burn-in}}(A', C', L). \end{aligned}$$

Substituting this into the regret bound proves the claim. $\square$

In particular, if we choose $\varepsilon \approx T^{-3/2}$, the discretization term becomes $O(1)$. Thus, whenever $Q_{\star}^{\gamma}(f, h; \varepsilon)$ is moderate, the dominant term in the nonlinear learning guarantee is the same stable-learning regret from Chapter 12, rather than the exponentially large lifted dimension N .

13.4 Bibliographic Remarks

The operator-theoretic viewpoint on nonlinear dynamics goes back to the work of [Koo31], who observed that nonlinear state evolution induces a linear evolution on observables. Modern Koopman theory, especially as it relates to model reduction, spectral analysis, and control, was further developed in [Mez05, Mez13]. Useful modern overviews include the review of [OR21], the survey of [BBKK22], and the volume [MMS20].

The finite-state discretization used in Section 13.1 is also closely related to the older literature on transfer operators and set-oriented numerics. In particular, our ε -net construction is more naturally viewed as an Ulam-type finite approximation than as a standard EDMD construction: one partitions the state space, represents the dynamics by a finite matrix on those cells, and then studies the resulting linear surrogate. See [Ula60], [Fro99], and the overview [Fro01] for this line of work.

On the data-driven side, [Sch10] popularized Dynamic Mode Decomposition (DMD) in fluid mechanics, and [TRL⁺14] clarified its operator-theoretic interpretation. Extended Dynamic Mode Decomposition (EDMD), which replaces the identity observable by a user-chosen dictionary, was introduced by [WKR15]. Related developments include generator-based methods such as gEDMD [KNP⁺20] and time-delay liftings [KKBK20]. These methods aim to approximate Koopman eigenfunctions or invariant subspaces directly from data.

A separate strand of the literature studies when nonlinear systems admit exact or nearly exact finite-dimensional Koopman representations. Finite-dimensional invariant subspaces for control were explored by [BBPK16], while learned Koopman coordinates and invariant subspaces were developed by [TKY17]. This distinction is conceptually important for the present chapter: exact finite-dimensional Koopman closures are special, whereas the discretization argument here gives a universal but approximate lifting.

For systems with inputs and control, operator-theoretic liftings lead naturally to linear or bilinear predictor models. Particularly relevant references are [PBK18] on Koopman operators with inputs and control, and [KM18] on linear predictors for nonlinear controlled systems and model predictive control. These works are natural points of comparison for the autonomous presentation adopted in this chapter.

Finally, the traditional alternative to the improper-learning viewpoint taken here is nonlinear system identification, which attempts to estimate the nonlinear transition map or latent state-space model directly from data. Good entry points to the classical literature are the perspective article of

[Lju10] and the state-space identification treatment of [SWN11]. More recently, there has been significant progress in providing non-asymptotic, finite-sample guarantees for learning nonlinear dynamical systems directly from a single trajectory. For instance, [FSR20] establish optimal sample complexity bounds for learning generalized nonlinear dynamics without relying on strict mixing assumptions. Similarly, [SO22] develop noise-sensitive, uniform convergence guarantees for learning nonlinear systems via gradient descent. The specific improper-learning reduction developed in this chapter, which combines a lifted linear approximation with the Luenberger framework from Chapter 12, is based on [DBH25].

13.5 Exercises

13.1. Consider the proof of Theorem 13.1. How does the dimension N of the approximating linear system scale with the dimension of the true state d_x ? Why does this curse of dimensionality not appear explicitly in the regret bounds derived from Chapter 12?

13.2. Theorem 13.1 assumed the dynamics f are 1-Lipschitz. Suppose instead that f is L -Lipschitz with $L > 1$.

1. Derive the recurrence satisfied by the tracking error ξ_t .
2. Show that ξ_t can grow exponentially with t .
3. What does this imply about the grid scale ε required to maintain low approximation error over a horizon T ?

13.3. Assume $x_0 > 0$, and consider the nonlinear system

$$x_{t+1} = x_t^2, \quad y_t = x_t.$$

1. Is this system linear in the state variable x ?
2. Show that the observable

$$g(x) = \log x$$

evolves linearly along the dynamics, in the sense that

$$g(x_{t+1}) = 2g(x_t).$$

3. Explain why this gives a one-dimensional Koopman-style lifting of the dynamics on the positive half-line.

Part IV

Online Control with Partial Observation

Chapter 14

Policy Classes for Partially Observed Systems

In previous chapters, we explored control and learning techniques in fully observed linear dynamical systems, where the system state is directly accessible to the controller. However, in many real-world scenarios, such as robotics, economics, and signal processing, the state is only partially observed, requiring the controller to infer system dynamics from noisy or incomplete observations.

This chapter focuses on partially observed linear dynamical systems, where the state is hidden, and only a linear projection of it is observed. For the remainder of this part of the text, we restrict ourselves to time-invariant systems and refer to the bibliographic section for extensions. The system follows the general form:

$$\begin{aligned}\mathbf{x}_{t+1} &= A\mathbf{x}_t + B\mathbf{u}_t + \mathbf{w}_t \\ \mathbf{y}_t &= C\mathbf{x}_t.\end{aligned}$$

Here, $\mathbf{x}_t$ represents the hidden system state, $\mathbf{u}_t$ the control input, $\mathbf{y}_t$ the observed output, and $\mathbf{w}_t$ the disturbance. Often, partially observed LDSs are defined with an additional noise term that corrupts the observations. Exercise 14.2 asks the reader to prove that even in such circumstances we may assume the previously stated form for partially observed systems without loss of any generality.

Unlike in the fully observed setting, where controllers can react based on the full state, partial observability introduces fundamental limitations on policy design.

In the partial observation setting, some of the control policy classes from Chapter 6 are not well defined. We then define policy classes and describe the relationship between them.

Not all control policies designed for fully observed systems extend naturally to partially observed settings. In this chapter, we systematically explore different policy classes, analyzing their expressivity, computational tractability, and suitability for partially observed LDS.

- **Linear Observation-Action Controllers** (or simply **Linear Controllers**): These controllers directly map observations to actions, but are highly inexpressive, as they ignore past observations and fail to capture hidden state dependencies.
- **Generalized Linear Observation-Action Controllers** (or **generalized linear controllers**): These extend linear controllers by incorporating the history of observations, significantly improving expressiveness.
- **Linear Dynamic Controllers (LDCs)**: These policies model internal system dynamics, making them more powerful than simple observation-based policies.
- **Disturbance Response Controllers (DRCs)**: Introduced as an alternative to state-based policies, DRCs act directly on a counterfactual signal, the “Nature’s y ’s”, which captures the response of the system that result by uniformly setting the control input to zero. We show that DRCs can approximate LDCs, indicating that they are a highly expressive and yet convex policy class.

We conclude by analyzing the relationships between these policy classes, showing how they fit into the broader framework of nonstochastic control with partial observation. This chapter lays the groundwork for online learning and regret minimization in partially observed LDS, which we will explore in the next chapter.

14.1 Linear Observation Action Controllers

Linear Observation-action controllers are the most natural generalization of linear controllers to the partially observed case, and are defined as follows.

Definition 14.1. *A linear observation-action controller $K \in \mathbb{R}^{d_u \times d_y}$ is a linear operator mapping observation to control as $\mathbf{u}_t^\pi = K\mathbf{y}_t$. We denote*

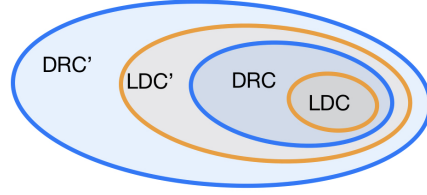


Figure 14.1: Schematic relationship between linear Disturbance-Response Control policies and Linear Dynamical Control policies of increasing dimensions (or number of parameters) for linear time invariant systems.

by Π_γ^L the set of all linear controllers bounded in the Frobenius norm by $\|K\| \leq \gamma$, which are also γ -stabilizing for given LDS.

Although this class of policies is natural to consider, it turns out to be very inexpressive. Even for optimal control with stochastic noise, the optimal policy cannot be approximated by a linear observation-action policy (see Exercise 14.6). Therefore, we now define more expressive and useful policies. However, linear controllers will be useful in simplifying proofs later on.

14.1.1 Generalized Linear Observation Action Controllers

We can generalize linear observation-action controllers to apply to a history of past observations. This additional generality, of taking a history of observations, greatly increases the expressive power of this policy class. The formal definition is given below.

Definition 14.2. Generalized linear observation-action controllers parameterized by matrices $K_{0:h}$ choose a control input as

$$\mathbf{u}_t^\pi = \sum_{i=0}^h K_i \mathbf{y}_{t-i}.$$

By $\Pi_{h,\gamma}^G$, we denote the class of generalized linear observation-action controllers that satisfy $\sum_{i=0}^h \|K_i\| \leq \gamma$ and are also γ -stabilizing.

Generalized observation-action policies can be seen as a special case of observation-action controllers using the technique of dimension lifting as follows. Consider any partially-observed system (A, B, C) and perturbation

sequence $\mathbf{w}_t$. Consider the *lifted* system $(\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$ given by

$$\mathbf{z}_{t+1} = \tilde{A}\mathbf{z}_t + \tilde{B}\mathbf{u}_t + \tilde{D}\mathbf{w}_t, \quad \tilde{\mathbf{y}}_t = \tilde{C}\mathbf{z}_t,$$

where

$$\tilde{A} = \begin{bmatrix} A & 0 & \dots & 0 & 0 \\ I & 0 & \dots & 0 & 0 \\ 0 & I & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & I & 0 \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} B_t \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

$$\tilde{C} = \begin{bmatrix} C & C & \dots & C \end{bmatrix}, \quad \tilde{D} = \begin{bmatrix} I \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

The following lemma relates the original and lifted system (for a proof, see exercise 14.3).

Lemma 14.3. *For any sequence of control inputs $\mathbf{u}_t$, it holds that $\tilde{\mathbf{y}}_t = [\mathbf{y}_t, \mathbf{y}_{t-1}, \dots, \mathbf{y}_{t-h}]$.*

The above lemma shows that generalized linear observation-action controllers are a special case of simple observation-action controllers such that $\mathbf{u}_t^\pi = [K_0 \dots K_h]\tilde{\mathbf{y}}_t$ for the lifted system.

14.2 Linear Dynamic Controllers

The most well-known class of controllers for partially observable linear dynamical systems is that of linear dynamical controllers due to its connection to Kalman filtering.

Definition 14.4 (Linear Dynamic Controllers for Partially Observed Systems). *A linear dynamic controller π has parameters (A_π, B_π, C_π) , and chooses a control at time t as*

$$\mathbf{u}_t^\pi = C_\pi \mathbf{s}_t, \quad \mathbf{s}_t = A_\pi \mathbf{s}_{t-1} + B_\pi \mathbf{y}_t.$$

We denote by Π_γ^{LDC} the class of all LDC that are γ -stabilizing as per Definition 2.4 and exercise 4.5, with parameters $\frac{\kappa}{\delta} \leq \gamma$ that satisfy

$$\forall i, \quad \|A_\pi^i\| \leq \kappa(1 - \delta)^i, \quad (\|B_\pi\| + \|C_\pi\|) \leq \kappa.$$

Note that LDCs are ε -approximated by generalized observation-action controllers. The proof is left as Exercise 14.4.

Lemma 14.5. *Consider a partially-observed time-invariant linear dynamical system $\{A, B, C\}$. The class $\Pi_{h', \gamma'}^G$ ε -approximates the class Π_γ^{LDC} for $h' = O(\gamma \log \frac{\gamma}{\varepsilon})$, $\gamma' = 2\gamma^3$.*

14.3 Disturbance Response Controllers

The most important policy class in the context of nonstochastic control is that of Disturbance Response Controllers, or DRC. These policies act on a signal called **Nature's y's**, denoted $\mathbf{y}_t^{\text{nat}}$, which is the would-be observation at time t assuming that all inputs to the system were zero from the beginning of time.

For ease of exposition, we only consider linear time invariant linear dynamical systems that are intrinsically stable. This allows us to ignore the stabilizing controller component of disturbance response control and greatly simplifies the presentation of the main ideas. The extension to time-varying systems is explored in the exercises.

Definition 14.6 (Disturbance Response Controller). *A disturbance response policy for stable systems $\pi_{M_{1:h}}$ is parameterized by matrices $M_{0:h} = [M_0, \dots, M_h]$. It outputs control $\mathbf{u}_t^\pi$ at time t according to the rule*

$$\mathbf{u}_t^\pi = \sum_{i=0}^h M_i \mathbf{y}_{t-i}^{\text{nat}},$$

where $\mathbf{y}_t^{\text{nat}}$ is given by the system response to the zero control at time t , i.e.

$$\mathbf{x}_{t+1}^{\text{nat}} = A\mathbf{x}_t^{\text{nat}} + \mathbf{w}_t = \sum_{i=0}^t A^i \mathbf{w}_{t-i}, \quad \mathbf{y}_t^{\text{nat}} = C\mathbf{x}_t^{\text{nat}}.$$

Denote by $\Pi_{h,\gamma}^R$ the set of all disturbance response policies that satisfy¹

$$\sum_{i=0}^h \|M_i\| \leq \gamma.$$

¹Recall that for this section, we restrict our discussion to stable systems. Otherwise, we would also ask the policy class to be γ -stabilizing, which can, for example, be achieved by superimposing a fixed γ -stabilizing linear dynamical controller.

If the perturbations are all zero, seemingly DRC policies produce zero control. Therefore, we could add a constant offset to the control that can be incorporated as in Definition 6.5.

The main advantage of this policy class, as opposed to previously considered policies in optimal control, is that they yield convex parametrization. Thus, this policy class allows for provably efficient optimization and algorithms that we explore in the next chapter.

We conclude with the following important lemma on an expression that is important to compute nature's y 's. The proof of this lemma follows directly from the definition and is left as Exercise 14.1.

Lemma 14.7. *The sequence $\{\mathbf{y}_t^{\text{nat}}\}$ can be iteratively computed using only the sequence of observations as*

$$\mathbf{y}_t^{\text{nat}} = \mathbf{y}_t - C \sum_{i=1}^t A^i B \mathbf{u}_{t-i}$$

Alternatively, the above computation can be carried out recursively as stated below starting with $\mathbf{z}_0 = \mathbf{0}$.

$$\begin{aligned} \mathbf{z}_{t+1} &= A\mathbf{z}_t + B\mathbf{u}_t, \\ \mathbf{y}_t^{\text{nat}} &= \mathbf{y}_t - C\mathbf{z}_t. \end{aligned}$$

We proceed to show that the class of DRC approximates the class of LDC.

14.3.1 Expressivity of Disturbance Response Controllers

A DRC always produces controls that ensure that the system remains state-bounded. This can be shown analogously to Lemma 6.6 (see Exercise 14.5).

Although DRCs are contained within the policy class of DACs, they are still very expressive. Note that a DAC policy can not be executed in a partially observed system from observations alone; even the knowledge of the entire observation sequence $\mathbf{y}_t$ may not uniquely determine $\mathbf{w}_t$.

We have already seen that LDCs are approximately captured by generalized linear controllers in Lemma 14.5. It can be shown that the class of DRCs with appropriate parameters ε -approximates the class of generalized linear policies for linear time-invariant systems. This observation in conjunction with Lemma 14.5 implies that DRCs also approximate LDCs.

To demonstrate that DRCs approximate generalized linear controllers, we instead prove DRCs approximate linear controllers, a weaker class of

policies. However, this is sufficient to conclude the stronger claim. To see this, first recall that generalized linear controllers acting on the last h states can be written as linear controllers on a lifted system with h states, as noted in Lemma 14.3. Together with the next lemma, this implies that a DRC on the lifted system can approximate generalized linear controllers on the original system. However, this is not quite what we promised. To wrap up the argument, one can verify that a DRC with history h' on the lifted system can be written as a DRC with history $h' + h$ on the original system.

Lemma 14.8. *Consider a γ -stable time-invariant linear dynamical system (A, B, C) . The class $\Pi_{h', \gamma'}^R$ ε -approximates the class Π_γ^L for the parameters $h' = O(\gamma \log \frac{\gamma}{\varepsilon})$, $\gamma' = O(\gamma^6)$.*

Recall that for fully observed LDS, the state-action pairs produced by execution of a linear policy can be written as linear functions of the perturbations. This observation allowed us to prove that DACs capture linear policies for fully observed systems. We prove an analogue for partially observed systems: The observation-action pairs produced by the execution of linear policies can be written as linear functions of nature's y 's. Thereafter, we argue that parameterizing the controllers to be linear in nature's y 's allows us to encompass linear observation-action policies.

Proof. We introduce an augmented partially-observed linear system that helps us write the observation-action pairs produced by a linear policy as a linear function of nature's y 's. In fact, we will shortly see that, for any sequence of control inputs, the augmented linear system produces the same sequence of observations as the original system. Thus, the new system has the same nature's y 's as the original system.

For any partially-observed system (A, B, C) and perturbation sequence $\mathbf{w}_t$ producing observations $\mathbf{y}_t$, construct a *lifted* system $(\tilde{A}, \tilde{B}, \tilde{C})$ given by

$$\begin{bmatrix} \mathbf{x}_{t+1}^{\text{nat}} \\ \Delta_{t+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & A \end{bmatrix} \begin{bmatrix} \mathbf{x}_t^{\text{nat}} \\ \Delta_t \end{bmatrix} + \begin{bmatrix} 0 \\ B \end{bmatrix} \mathbf{u}_t + \begin{bmatrix} I \\ 0 \end{bmatrix} \mathbf{w}_t, \quad \tilde{\mathbf{y}}_t = \begin{bmatrix} C & C \end{bmatrix} \begin{bmatrix} \mathbf{x}_t^{\text{nat}} \\ \Delta_t \end{bmatrix}.$$

Fix any sequence of control inputs $\mathbf{u}_t$. Let $\mathbf{x}_t, \mathbf{y}_t$ be the states and observations produced by the original partially-observed system. The equivalents for the augmented system are $[\mathbf{x}_t^{\text{nat}}, \Delta_t]^\top, \tilde{\mathbf{y}}_t$, as we have described above. Now we can verify for all time t that $\tilde{\mathbf{y}}_t = \mathbf{y}_t$ and $\mathbf{x}_t = \mathbf{x}_t^{\text{nat}} + \Delta_t$.

Since this new system has the same input-output behavior as the original system, we might equivalently investigate how the execution of a linear controller π affects the augmented system. Substituting $\mathbf{u}_t^\pi = K\mathbf{y}_t^\pi =$

$KC(\mathbf{x}_t^{\text{nat}} + \Delta_t^\pi)$, the state-action sequence produced by a linear policy $\mathbf{u}_t^\pi = K\mathbf{y}_t^\pi$ can be written as

$$\begin{bmatrix} \mathbf{x}_{t+1}^{\text{nat}} \\ \Delta_{t+1}^\pi \end{bmatrix} = \begin{bmatrix} A & 0 \\ BKC & A + BKC \end{bmatrix} \begin{bmatrix} \mathbf{x}_t^{\text{nat}} \\ \Delta_t^\pi \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix} \mathbf{w}_t, \quad \begin{bmatrix} \mathbf{y}_t^\pi \\ \mathbf{u}_t^\pi \end{bmatrix} = \begin{bmatrix} C & C \\ KC & KC \end{bmatrix} \begin{bmatrix} \mathbf{x}_t^{\text{nat}} \\ \Delta_t^\pi \end{bmatrix},$$

or using that $\mathbf{y}_t^{\text{nat}} = C\mathbf{x}_t^{\text{nat}}$, equivalently as

$$\Delta_{t+1}^\pi = (A + BKC)\Delta_t^\pi + BK\mathbf{y}_t^{\text{nat}}, \quad \mathbf{y}_t^\pi = C\Delta_t^\pi + \mathbf{y}_t^{\text{nat}}, \quad \mathbf{u}_t^\pi = KC\Delta_t^\pi + K\mathbf{y}_t^{\text{nat}}.$$

Now, define $M_i = KC(A + BKC)^{i-1}BK$ for all positive $i \in [h']$ and $M_0 = K$. These parameters specify our DRC. Let $\mathbf{y}_t, \mathbf{u}_t$ be the observation-action sequence produced by this DRC. We have

$$\begin{aligned} \mathbf{u}_t^\pi &= K\mathbf{y}_t^{\text{nat}} + KC\Delta_t^\pi \\ &= K\mathbf{y}_t^{\text{nat}} + KCBK\mathbf{y}_{t-1}^{\text{nat}} + KC(A + BKC)\Delta_{t-1}^\pi \\ &= K\mathbf{y}_t^{\text{nat}} + \sum_{i>1} KC(A + BKC)^{i-1}BK\mathbf{y}_{t-i}^{\text{nat}} \\ &= \sum_{i=0}^{h'} M_i\mathbf{y}_{t-i}^{\text{nat}} + \sum_{i>H'} KC(A + BKC)^{i-1}BK\mathbf{y}_{t-i}^{\text{nat}} \\ &= \mathbf{u}_t + Z_t. \end{aligned}$$

Given γ -stabilizability of the linear controller K and γ -intrinsic stability of the of the system (A, B, C) , we have that for some κ, δ such that $\frac{\kappa}{\delta} \leq \gamma$,

$$\begin{aligned} \|Z_t\| &\leq \sum_{i>h'} \|KC(A + BKC)^{i-1}BK\| \|\mathbf{y}_{t-i}^{\text{nat}}\| \\ &\leq \sum_{i>h'} \|KC(A + BKC)^{i-1}BK\| \left(\sum_{j>1} \|CA^{j-1}\mathbf{w}_{t-i-j}\| \right) \\ &\leq \sum_{i>h'} \|K\|^2 \kappa (1 - \delta)^{i-1} \sum_{j>1} \kappa (1 - \delta)^j \\ &\leq \frac{\kappa^4}{\delta} \sum_{i>h'} (1 - \delta)^{i-1} \leq \frac{\kappa^4}{\delta^2} (1 - \delta)^{h'} \leq \frac{\kappa^4}{\delta^2} e^{-\delta h'} \end{aligned}$$

Therefore, $\|\mathbf{u}_t - \mathbf{u}_t^\pi\| \leq \varepsilon$ by the appropriate choice of h' . Once again using the *intrinsic* stability of the LTI (A, B, C) , we have

$$\begin{aligned} \|\mathbf{y}_t^\pi - \mathbf{y}_t\| &= \left\| \sum_{i=1}^t C A^{i-1} B (\mathbf{u}_{t-i}^\pi - \mathbf{u}_{t-i}) \right\| \\ &\leq \left\| \sum_{i=0}^t C A^i B \right\| \max_{i \leq t} \|\mathbf{u}_i^\pi - \mathbf{u}_i\| \leq \frac{\kappa}{\delta} \varepsilon. \end{aligned}$$

The approximation in terms of observations and controls implies ε -approximation of the policy class for any Lipschitz cost sequence, concluding the proof. $\square$

14.4 Summary

In this chapter, we extended our study of control to the partially observed setting, where the controller no longer has direct access to the system state but must instead rely on noisy or incomplete observations. This shift fundamentally changes the nature of the control problem, requiring the design of policies that can infer and act on the basis of limited information.

We introduced a hierarchy of policy classes, starting with Linear Observation-Action Controllers, which directly map observations to actions, and progressing to generalized linear policies and Linear Dynamic Controllers (LDCs), which incorporate history to improve decision-making. This increasing expressiveness highlighted a key limitation: memoryless policies are insufficient for optimal control, making structured state-dependent controllers necessary.

A major contribution of this chapter was the introduction of Disturbance Response Controllers (DRCs). Instead of explicitly estimating the state of the system, DRCs act directly on a counterfactual signal, the “Nature’s y ’s”, capturing what the system would have observed in the absence of control inputs. We established that DRCs are a convex relaxation of LDCs, making them computationally attractive while maintaining strong expressive power. This result provides a principled approach to designing policies that are efficient and robust in partially observed environments.

This chapter also set the stage for online nonstochastic control under partial observability, which we will explore in the next chapter. By establishing policy expressiveness results and identifying convex approximations for optimal control, we have built the necessary framework for developing learning-based algorithms that operate effectively in uncertain and adversarial settings.

14.5 Bibliographic Remarks

The study of control in partially observed linear dynamical systems has a long history, with foundational contributions dating back to optimal estimation and control theories. This section provides an overview of key references related to the topics discussed in this chapter, including classical works, expressivity of policy classes, convex relaxations, and connections to online learning.

The problem of control under partial observation was first rigorously formulated in the context of the Linear Quadratic Gaussian (LQG) regulator, which combines Kalman filtering with optimal control [AM79]. The certainty equivalence principle, which states that optimal controllers can be designed assuming a perfect state estimate, was shown to have limitations in more general partially-observed settings [Won68].

This chapter introduced a hierarchy of policy classes, including linear observation-action controllers, generalized linear controllers, Linear Dynamic Controllers (LDCs), and Disturbance Response Controllers (DRCs). The concept of disturbance response controllers was inspired by parameterizations used in robust control, particularly the Youla-Kucera parameterization [YJB76].

Convex relaxations have been a powerful tool for simplifying control design in partially observed LDS. We have surveyed convex relaxations for learning in dynamical systems in the previous part of this book. Recently, spectral methods for control [AHL⁺18] have provided an alternative approach to convex relaxation, utilizing spectral filtering to bypass direct system identification.

The Disturbance Response Parametrization was introduced in [SSH20] as an analogue of DAC to control partially-observed systems. It can be viewed as a succinct state-space representation of the Youla parameterization [YJB76]. For time-varying dynamics, the power of various policy classes was contrasted in [MGSH21].

14.6 Exercises

14.1. Prove Lemma 14.7, stating that the sequence $\{\mathbf{y}_t^{\text{nat}}\}$ can be iteratively computed using only the sequence of observations as

$$\mathbf{y}_t^{\text{nat}} = \mathbf{y}_t - C \sum_{i=1}^t A^i B \mathbf{u}_{t-i}$$

Alternatively, the above computation can be carried out recursively as stated below starting with $\mathbf{z}_0 = \mathbf{0}$.

$$\begin{aligned} \mathbf{z}_{t+1} &= A\mathbf{z}_t + B\mathbf{u}_t, \\ \mathbf{y}_t^{\text{nat}} &= \mathbf{y}_t - C\mathbf{z}_t. \end{aligned}$$

14.2. Consider a seemingly more general definition of a partially observed linear dynamical system, given below:

$$\begin{aligned} \mathbf{x}_{t+1} &= A_t \mathbf{x}_t + B_t \mathbf{u}_t + \mathbf{w}_t, \\ \mathbf{y}_t &= C_t \mathbf{x}_t + \mathbf{v}_t. \end{aligned}$$

Construct a linear dynamical system parameterized via matrices (A'_t, B'_t, C'_t) and perturbations $\mathbf{w}'_t$ with a hidden state $\mathbf{z}_t$ of the form

$$\begin{aligned} \mathbf{x}_{t+1} &= A'_t \mathbf{z}_t + B'_t \mathbf{u}_t + \mathbf{w}'_t, \\ \mathbf{y}'_t &= C'_t \mathbf{z}_t \end{aligned}$$

such that for any sequence of $\mathbf{u}_t$, $\mathbf{y}_t = \mathbf{y}'_t$ holds uniformly over time. Additionally, verify that there exists a construction of $\mathbf{z}_t$ where $\mathbf{w}'_t$ are i.i.d. as long as $(\mathbf{w}_t, \mathbf{v}_t)$ are i.i.d. across time steps.

Hint: Consider the choice $\mathbf{w}'_t = D'_t \begin{bmatrix} \mathbf{w}_t \\ \mathbf{v}_t \end{bmatrix}$ for some D'_t . Also, consider using a hidden state $\mathbf{z}_t$ with a larger number of coordinates than $\mathbf{x}_t$.

14.3.

1. Prove Lemma 14.3.
2. Extend the lemma and prove it for time-varying linear dynamical systems.

14.4. Prove Lemma 14.5.

14.5. For any γ -stable partially observed system, prove that any DRC policy in $\Pi_{H,\gamma}^D$ is γ^2 -stabilizing.

14.6. Construct a partially observed linear dynamical system subject to Gaussian i.i.d. perturbations with quadratic costs such that simultaneously it is true that (a) there exists a gearlized linear policy that incurs zero (or at most constant)

aggregate cost, (b) yet any linear observation-action policy incurs $\Omega(T)$ aggregate cost when run for T time steps.

Hint: Consider a 3-dimensional hidden state so that u_t can be chosen as a function of y_{t-1} to ensure zero cost at each time step.

Chapter 15

Online Nonstochastic Control with Partial Observation

In the previous chapter, we introduced Disturbance Response Controllers (DRCs) as a structured class of policies for controlling partially observed linear dynamical systems (PO-LDS). Unlike classical observer-based controllers, which rely on explicit state estimation, DRCs act directly on Nature's y 's, the system's counterfactual response in the absence of control. Although DRCs offer a powerful alternative to state-based policies, we have not yet integrated them into an online learning algorithm, leaving open the question of how to adapt them dynamically in uncertain environments.

This chapter takes the next step by introducing an online learning algorithm for optimizing DRCs in real time. We introduce the Gradient Response Controller (GRC), which efficiently updates DRC policies using online convex optimization techniques. The GRC algorithm enables controllers to:

- Learn from past disturbances and actions, refining policy parameters without requiring full system identification.
- Handle adversarial disturbances, extending the policy regret framework to the partially observed setting.
- Optimize control decisions dynamically, adapting to system behavior as new observations arrive.

Unlike previous approaches that relied on fixed policy classes, GRC provides an adaptive control mechanism that bridges the gap between structured controllers and learning-based methods. A key focus of this chapter is to extend

policy regret minimization from the fully observed case to partial observability, introducing new challenges and solutions in learning-based control.

To simplify the theoretical analysis, we assume system stability, allowing us to focus on the core learning problem without additional stabilization constraints. However, we discuss possible extensions to stabilizable systems in the Bibliographic Remarks section.

15.1 The Gradient Response Controller

Partial observation requires modification to the Gradient Perturbation Controller we have studied in the previous chapter. The Gradient Response Controller algorithm is described in figure (9), for linear time-invariant dynamical systems with partial observation. Similarly to GPC, the main idea is to learn the parameterization of a disturbance response control policy using known online convex optimization algorithms, such as online gradient descent. We henceforth prove that this parameterization is convex, allowing us to prove the main regret bound.

In algorithm (9) we denote $\mathbf{y}_t(M_{1:h})$ as the observation arising from playing the DRC policy $M_{1:h}$ from the beginning of time, and the same for $\mathbf{u}_t(M_{1:h})$. Although it may be impossible to determine $\mathbf{w}_t$, especially if C has fewer rows than columns, from the observations $\mathbf{y}_t$, $\mathbf{y}_t(M_{1:h})$ can be determined from $\mathbf{y}_t^{\text{nat}}$ alone by the application of Lemma 14.7, as

$$\mathbf{y}_t(M_{1:h}) = \mathbf{y}_t^{\text{nat}} + C \sum_{i=1}^t A^i B \mathbf{u}_{t-i}(M_{1:h}).$$

Algorithm 9 Gradient Response Controller(GRC)

- 1: Input: system $\{A, B, C\}$, h , η , initialization $M_{0:h}^1 \in \mathcal{K}$.
- 2: **for** $t = 1 \dots T$ **do**
- 3: Use Control $\mathbf{u}_t = \sum_{i=0}^h M_i^t \mathbf{y}_{t-i}^{\text{nat}}$
- 4: Observe $\mathbf{y}_{t+1}$, compute for $t + 1$:

$$\mathbf{y}_t^{\text{nat}} = \mathbf{y}_t - C \sum_{i=0}^t A^i B \mathbf{u}_{t-i}.$$

- 5: Construct loss $\ell_t(M_{0:h}) = c_t(\mathbf{y}_t(M_{0:h}), \mathbf{u}_t(M_{0:h}))$
 - 6: Update $M_{0:h}^{t+1} \leftarrow \prod_{\mathcal{K}} [M_{0:h}^t - \eta \nabla \ell_t(M_{0:h}^t)]$
 - 7: **end for**
-

The GRC algorithm comes with a finite-time performance guarantee: it guarantees vanishing worst-case regret versus the best disturbance response control policy in hindsight. Following the relationship between disturbance response control and other policies studied in Section 14.3.1, this implies vanishing regret versus linear dynamical controllers. The formal statement is given as follows.

Theorem 15.1. *Assuming that*

- a The costs c_t are convex, bounded and have bounded gradients with respect to the arguments $\mathbf{y}_t$ and $\mathbf{u}_t$.*
- b The diameter of the constraint set for the parameters $M_{0:h}$ is bounded by D .*
- c The matrices $\{A, B, C\}$ have bounded ℓ_2 norms.*
- d The linear dynamical system $\{A, B, C\}$ is γ -stable.*

Then the GRC algorithm (9) ensures that

$$\max_{\mathbf{w}_{1:T}: \|\mathbf{w}_t\| \leq 1} \left(\sum_{t=1}^T c_t(\mathbf{y}_t, \mathbf{u}_t) - \min_{\pi \in \Pi^R} \sum_{t=1}^T c_t(\mathbf{y}_t^\pi, \mathbf{u}_t^\pi) \right) \leq \mathcal{O}(LGD\gamma^3 h \sqrt{T}).$$

Furthermore, the time complexity of each loop of the algorithm is polynomial in the number of system parameters and logarithmic in T .

The proof is largely analogous to the regret bound of the GPC presented in Theorem 7.2.

Proof. According to Theorem 6.1, the online gradient descent algorithm for convex loss functions ensures that the regret is bounded as a sublinear function of the number of iterations. However, to directly apply the OGD algorithm to our setting, we need to make sure that the following conditions hold true:

- The loss function should be a convex function in the variables $M_{0:h}$.
- We must ensure that optimizing the loss function ℓ_t as a function of parameters $M_{0:h}$ is similar to optimizing the cost c_t as a function of observation and control.

First, we show that the loss function is convex with respect to the variables $M_{0:h}$. This follows since the states and the controls are linear transformations of the variables.

Lemma 15.2. *The loss functions ℓ_t are convex in the variables $M_{0:h}$.*

Proof. Using $\mathbf{y}_t$ to denote $\mathbf{y}_t(M_{0:h})$ (and similarly for $\mathbf{u}_t$), the loss function ℓ_t is given by

$$\ell_t(M_{0:h}) = c_t(\mathbf{y}_t, \mathbf{u}_t).$$

Since the cost c_t is a convex function with respect to its arguments, we simply need to show that $\mathbf{y}_t$ and $\mathbf{u}_t$ depend linearly on $M_{0:h}$. Using Lemma 14.7,

$$\begin{aligned} \mathbf{u}_t &= \sum_{i=0}^h M_i \mathbf{y}_{t-i}^{\text{nat}}, \\ \mathbf{y}_t &= \mathbf{y}_t^{\text{nat}} + C \sum_{i=1}^t A^i B \mathbf{u}_{t-i} \\ &= \mathbf{y}_t^{\text{nat}} + C \sum_{i=1}^t A^i B \sum_{j=0}^h M_j \mathbf{y}_{t-i-j}^{\text{nat}} \end{aligned}$$

which is a linear function of the variables. Thus, we have shown that $\mathbf{y}_t$ and $\mathbf{u}_t$ are linear transformations in $M_{0:h}$ and, hence, the loss function is convex in $M_{0:h}$. $\square$

Next, we use the following lemma.

Lemma 15.3. *For all t , $|l_t(M_{0:h}^t) - c_t(\mathbf{y}_t, \mathbf{u}_t)| \leq \frac{L\gamma^3 ChD}{\sqrt{T}}$.*

The regret bound of GRC can be derived from these lemmas as follows. Note that G is an upper bound on the gradients of ℓ_t as a function of the parameters $M_{0:h}$, while L is the Lipschitz constant of c_t as a function of the state and control.

$$\begin{aligned} & \sum_{t=1}^T c_t(\mathbf{y}_t, \mathbf{u}_t) - \min_{\pi \in \Pi^R} \sum_{t=1}^T c_t(\mathbf{y}_t^\pi, \mathbf{u}_t^\pi) \\ & \leq \sum_{t=1}^T l_t(M_{1:h}^t) - \min_{\pi \in \Pi^R} l_t(M_{1:h}^\pi) + T \times \frac{L\gamma^3 ChD}{\sqrt{T}} && \text{Lemma 15.3} \\ & \leq 2GD\sqrt{T} + \sqrt{T}L\gamma^3 ChD && \text{Theorem 6.1} \\ & \leq O(GDL\gamma^3 h\sqrt{T}). \end{aligned}$$

$\square$

15.1.1 Finishing up the Regret Bound

Proof of Lemma 15.3. We first note that by the choice of step size by the Online Gradient Descent algorithm 3, as per Theorem 6.1, we have $\eta = \frac{D}{G\sqrt{T}}$. Thus,

$$\|M_{0:h}^t - M_{0:h}^{t-i}\| \leq \sum_{s=t-i+1}^t \|M_{0:h}^s - M_{0:h}^{s-1}\| \leq i\eta G = \frac{iD}{\sqrt{T}}.$$

Now, we use the fact that the iterates of OGD $M_{1:h}^t$ move slowly to establish that $\mathbf{y}_t$ and $\mathbf{y}_t(M_{0:h}^t)$ are close in value. Before that, by definition, we have $\mathbf{u}_t(M_{0:h}^t) = \mathbf{u}_t$ since the sequence $\mathbf{y}_t^{\text{nat}}$ is determined independently of the choice of the policy that is being executed. Due to the γ -stability of the underlying system, for any $i > 0$, $\|CA^iB\| \leq \kappa(1-\delta)^i$ for some $\kappa > 0, \gamma < 1$. Using Lemma 14.7, we have for some $C > 0$ that

$$\begin{aligned} \|\mathbf{y}_t(M_{0:h}^t) - \mathbf{y}_t\| &= \left\| C \sum_{i=1}^t [A_{i-1}B(\mathbf{u}_{t-i}(M_{0:h}^t) - \mathbf{u}_{t-i})] \right\| \\ &\leq \sum_{i=1}^t \|CA^{i-1}B\| \|(\mathbf{u}_{t-i}(M_{0:h}^t) - \mathbf{u}_{t-i})\| \\ &\leq Ch\kappa \sum_{i=1}^t (1-\delta)^i \frac{i}{\sqrt{T}} \max_j \|\mathbf{y}_j^{\text{nat}}\| \\ &\leq \frac{Ch\kappa D}{\delta^2\sqrt{T}} \max_j \|\mathbf{y}_j^{\text{nat}}\| \leq \frac{ChD\gamma^3}{\sqrt{T}} \end{aligned}$$

Here we used the fact that $\|\mathbf{y}_t^{\text{nat}}\| \leq \gamma$, which is left as an exercise.

The control depends only on the parameters of the current iteration, and we have $\mathbf{u}_t = \mathbf{u}_t(M_{1:h}^t)$. By definition, $l_t(M_{0:h}^t) = c_t(\mathbf{y}_t(M_{0:h}^t), \mathbf{u}_t(M_{0:h}^t))$. Thus, we have

$$\begin{aligned} l_t(M_{0:h}^t) - c_t(\mathbf{y}_t, \mathbf{u}_t) &= c_t(\mathbf{y}_t(M_{0:h}^t), \mathbf{u}_t(M_{0:h}^t)) - c_t(\mathbf{y}_t, \mathbf{u}_t) \\ &\leq L \cdot \|\mathbf{y}_t(M_{1:h}^t) - \mathbf{y}_t\| \leq \frac{L\gamma^3 ChD}{\sqrt{T}}. \end{aligned}$$

□

15.2 Extension to Time Varying Systems

The GRC algorithm can be extended to time-varying dynamics as we explain in this section. The linear dynamical system at time t is denoted by

the system matrices A_t, B_t, C_t , and we assume that they are known to the controller. We further assume that these dynamical systems are all γ -stable. The algorithm is formally spelled out in figure 10.

Algorithm 10 Gradient Response Controller(GRC)

- 1: Input: sequence $\{A_t, B_t, C_t\}$, h, η , initialization $M_{0:h}^1 \in \mathcal{K}$.
- 2: **for** $t = 1 \dots T$ **do**
- 3: Use Control $\mathbf{u}_t = \sum_{i=0}^h M_i^t \mathbf{y}_{t-i}^{\text{nat}}$
- 4: Observe $\mathbf{y}_{t+1}$, compute for $t+1$:

$$\mathbf{y}_t^{\text{nat}} = \mathbf{y}_t - C_t \sum_{i=0}^t \left[\prod_{j=0}^i A_{t-j} \right] B_{t-i} \mathbf{u}_{t-i}.$$

- 5: Construct loss $\ell_t(M_{0:h}) = c_t(\mathbf{y}_t(M_{0:h}), \mathbf{u}_t(M_{0:h}))$
 Update $M_{0:h}^{t+1} \leftarrow \prod_{\mathcal{K}} [M_{0:h}^t - \eta \nabla \ell_t(M_{0:h}^t)]$
 - 6: **end for**
-

It is possible to prove a similar regret bound to the one we proved for linear time-invariant systems. An important step is to note that the observation $\mathbf{y}_t(M_{1:h})$ can be determined from $\mathbf{y}_t^{\text{nat}}$ alone by the application of Lemma 14.7, as

$$\mathbf{y}_t(M_{1:h}) = \mathbf{y}_t^{\text{nat}} + C_t \sum_{i=1}^t \left[\prod_{j=1}^{i-1} A_{t-j} \right] B_{t-i} \mathbf{u}_{t-i}(M_{1:h}).$$

15.3 Conclusion

In this chapter, we extended the framework of online nonstochastic control to the partially observed setting, where the controller must make decisions without direct access to the system state. Unlike the fully observed case, where state-feedback policies can be applied directly, partial observability introduces fundamental challenges, requiring policies that infer system behavior from past observations and actions.

Building on the Disturbance Response Controller (DRC) policies introduced in the previous chapter, we developed the Gradient Response Controller (GRC), an online learning algorithm that optimizes DRC policies in real time.

The results in this chapter demonstrate that adaptive control under partial observability is possible without explicit state estimation, bridging the

gap between classical observer-based controllers and modern learning-based methods. Using regret minimization techniques, we provide a robust and flexible approach that applies even in adversarial and uncertain environments.

15.4 Bibliographic Remarks

Online nonstochastic control with partial observability was studied in [SSH20], who defined the Gradient Response Controller as well as the policy class of Disturbance Response Controllers.

The setting was further extended to time-varying linear dynamics in [MGSH21], where the expressive power of different classes of policy was also studied.

The analogue of online nonstochastic control in the stochastic noise setting is called the Linear Quadratic Gaussian problem, and it is further limited by assuming quadratic noise. This classical problem in optimal control was proposed in Kalman’s work [Kal60]. Regret bounds for controlling unknown partially-observed systems were studied in [LAHA20b, LAHA20c, MTR19]. [LAHA20a], which use DRC policies even in the context of stochastic control, obtain rate-optimal logarithmic bounds for the LQG problem.

In [SNH23], the task of controlling a partially-observed system subject to semi-adversarial noise under bandit feedback was considered. The tight rates for quadratic control with bandit feedback, in the fully adversarial model, were obtained in [SSNH24], and more general cost functions in the bandit setting were considered in [SL24].

15.5 Exercises

15.1. In this exercise, we consider the computation of nature's y 's for time-varying systems. Let

$$\begin{aligned}\mathbf{x}_{t+1}^{\text{nat}} &= A_t \mathbf{x}_t^{\text{nat}} + \mathbf{w}_t = \sum_{i=0}^t \left[\prod_{j=0}^{i-1} A_{t-j} \right] \mathbf{w}_{t-i} = \Phi_t \mathbf{w}_{1:t}. \\ \mathbf{y}_{t+1}^{\text{nat}} &= C_{t+1} \mathbf{x}_{t+1}^{\text{nat}} = C_{t+1} \Phi_t \mathbf{w}_{1:t}.\end{aligned}$$

Here Φ_t is the linear operator such that $\Phi_t(t-i) = \prod_{j=0}^{i-1} A_{t-j}$. Prove that The sequence $\{\mathbf{y}_t^{\text{nat}}\}$ can be iteratively computed using only the sequence of observations as

$$\mathbf{y}_t^{\text{nat}} = \mathbf{y}_t - C_t \sum_{i=1}^t \left[\prod_{j=1}^{i-1} A_{t-j} \right] B_{t-i} \mathbf{u}_{t-i}$$

Alternatively, the above computation can be carried out recursively as stated below starting with $\mathbf{z}_0 = \mathbf{0}$.

$$\begin{aligned}\mathbf{z}_{t+1} &= A_t \mathbf{z}_t + B_t \mathbf{u}_t, \\ \mathbf{y}_t^{\text{nat}} &= \mathbf{y}_t - C_t \mathbf{z}_t.\end{aligned}$$

15.2. Consider a partially observed linear dynamical system that is γ -stable in the absence of control inputs. Prove that when a DRC policy $\pi \in \Pi_{H,\gamma}^R$ is executed on such a dynamical system, the resulting system is γ^2 -stable.

15.3. Prove that the optimal controller that minimizes the aggregate cost for a partially observed linear dynamical system assuming that the perturbations are stochastic is a linear dynamical controller.

15.4. Prove that the regret of the GRC controller for time-varying linear dynamical systems is bounded by $O(\sqrt{T})$.

Hint: Follow the proof of Theorem 15.1 and modify as required.

Appendix A

Second Order Online Learning with the Square Loss

This appendix presents second order methods for the special case of online convex optimization with the square loss. The first algorithm is a specialization of the Online Newton Step (ONS) algorithm for the square loss. ONS operates strictly within the standard Online Convex Optimization (OCO) protocol, meaning no feature vector is revealed to the learner prior to making a decision. In addition, ONS gracefully handles arbitrary constrained convex sets for the learned parameters. The proof follows the same potential-based argument as in [Haz16, Theorem 4.5], but exploits the exact quadratic structure of square loss.

The second method we describe is the classical VAW algorithm, which does not strictly fall within the framework of online convex optimization due to two limitations. First, it requires knowing the feature vector before prediction, and secondly, it does not allow for constraints. However, in many control scenarios studied in this text, these limitations are acceptable, and in return it offers superior regret bounds in some respects.

A.1 Setting

We consider the standard Online Convex Optimization (OCO) protocol. At each round $t = 1, \dots, T$:

- The learner chooses a parameter $\mathbf{x}_t \in \mathcal{K} \subseteq \mathbb{R}^d$, where $\mathcal{K}$ is a closed convex set with diameter D .

- The environment reveals a feature vector $\mathbf{a}_t \in \mathbb{R}^d$ such that $\|\mathbf{a}_t\| \leq R$, and a true label $b_t \in \mathbb{R}$, defining the loss function $\ell_t(\mathbf{x}) = \frac{1}{2}(\mathbf{x}^\top \mathbf{a}_t - b_t)^2$.
- The learner suffers loss $\ell_t(\mathbf{x}_t)$. Assume that the losses suffered by the learner are bounded by a constant $C > 0$:

$$\max_{t=1,\dots,T} \ell_t(\mathbf{x}_t) = \max_t \frac{1}{2}(\mathbf{x}_t^\top \mathbf{a}_t - b_t)^2 \leq C.$$

Define regret against $\mathbf{x}^* \in \mathcal{K}$:

$$\text{regret}_T(\mathbf{x}^*) = \sum_{t=1}^T \ell_t(\mathbf{x}_t) - \ell_t(\mathbf{x}^*).$$

A.2 Algorithm: ONS for Square Loss

Algorithm 11 ONS for Square Loss

- 1: Input: regularization $\lambda > 0$
- 2: Initialize $A_0 = \lambda I_d$, $\mathbf{x}_1 \in \mathcal{K}$
- 3: **for** $t = 1$ to T **do**
- 4: Play $\mathbf{x}_t$
- 5: Observe $\mathbf{a}_t, b_t$ and suffer loss $\ell_t(\mathbf{x}_t) = \frac{1}{2}(\mathbf{x}_t^\top \mathbf{a}_t - b_t)^2$
- 6: Compute gradient $\nabla_t = \nabla \ell_t(\mathbf{x}_t) = (\mathbf{x}_t^\top \mathbf{a}_t - b_t)\mathbf{a}_t$
- 7: Update curvature matrix:

$$A_t = A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top$$

- 8: Update parameter:

$$\mathbf{x}_{t+1} = \prod_{\mathcal{K}}^{A_t} (\mathbf{x}_t - A_t^{-1} \nabla_t)$$

- 9: **end for**
-

A.3 Regret Bound

Theorem A.1 (Logarithmic regret for square loss). *For Algorithm 11, assuming losses are bounded by C and $\mathcal{K}$ has diameter D , for all $\mathbf{x}^* \in \mathcal{K}$,*

$$\text{regret}_T(\mathbf{x}^*) \leq \frac{\lambda}{2} D^2 + C \log \det \left(I_d + \frac{1}{\lambda} \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top \right).$$

In particular, setting $\lambda = 1/D^2$, and $d, C, T, R, D > 2$ we get

$$\text{regret}_T(\mathbf{x}^*) \leq \frac{\lambda}{2} D^2 + dC \log \left(1 + \frac{TR^2}{\lambda} \right) \leq 4dC \log(TRD).$$

Proof. Fix $\mathbf{x}^* \in \mathcal{K}$. Let $r_t = \mathbf{x}_t^\top \mathbf{a}_t - b_t$ denote the prediction residual. For square loss we have the exact identity

$$\ell_t(\mathbf{x}_t) - \ell_t(\mathbf{x}^*) = \nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) - \frac{1}{2} ((\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t)^2.$$

Indeed, expanding the difference in losses yields

$$\begin{aligned} \ell_t(\mathbf{x}_t) - \ell_t(\mathbf{x}^*) &= \frac{1}{2} \left(r_t^2 - (r_t - (\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t)^2 \right) \\ &= r_t (\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t - \frac{1}{2} ((\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t)^2 \\ &= \nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) - \frac{1}{2} ((\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t)^2. \end{aligned}$$

We define the potential $\Phi_t = \|\mathbf{x}_t - \mathbf{x}^*\|_{A_{t-1}}^2$. By the generalized Pythagorean theorem for projection onto a convex set in the A_t -norm,

$$\|\mathbf{x}_{t+1} - \mathbf{x}^*\|_{A_t}^2 \leq \|\mathbf{x}_t - A_t^{-1} \nabla_t - \mathbf{x}^*\|_{A_t}^2.$$

Expanding the right-hand side gives

$$\|\mathbf{x}_t - A_t^{-1} \nabla_t - \mathbf{x}^*\|_{A_t}^2 = \|\mathbf{x}_t - \mathbf{x}^*\|_{A_t}^2 - 2 \nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) + \nabla_t^\top A_t^{-1} \nabla_t.$$

Rearranging this inequality to bound the linear term yields

$$\nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) \leq \frac{1}{2} (\|\mathbf{x}_t - \mathbf{x}^*\|_{A_t}^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|_{A_t}^2) + \frac{1}{2} \nabla_t^\top A_t^{-1} \nabla_t.$$

Since the curvature matrix is updated via $A_t = A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top$, we can relate the norms as

$$\|\mathbf{x}_t - \mathbf{x}^*\|_{A_t}^2 = \|\mathbf{x}_t - \mathbf{x}^*\|_{A_{t-1}}^2 + ((\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t)^2.$$

Plugging this expansion into our previous inequality and moving the resulting quadratic term to the left-hand side gives

$$\nabla_t^\top (\mathbf{x}_t - \mathbf{x}^*) - \frac{1}{2} ((\mathbf{x}_t - \mathbf{x}^*)^\top \mathbf{a}_t)^2 \leq \frac{1}{2} \left(\|\mathbf{x}_t - \mathbf{x}^*\|_{A_{t-1}}^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|_{A_t}^2 \right) + \frac{1}{2} \nabla_t^\top A_t^{-1} \nabla_t.$$

By the exact identity established above, the left-hand side is exactly the instantaneous regret $\ell_t(\mathbf{x}_t) - \ell_t(\mathbf{x}^*)$. Summing over $t = 1$ to T telescopes the potential terms:

$$\text{regret}_T(\mathbf{x}^*) \leq \frac{1}{2} \|\mathbf{x}_1 - \mathbf{x}^*\|_{A_0}^2 + \frac{1}{2} \sum_{t=1}^T \nabla_t^\top A_t^{-1} \nabla_t.$$

Since $A_0 = \lambda I_d$, we have $\frac{1}{2} \|\mathbf{x}_1 - \mathbf{x}^*\|_{A_0}^2 = \frac{\lambda}{2} \|\mathbf{x}^* - \mathbf{x}_1\|_2^2 \leq \frac{\lambda}{2} D^2$, where the last inequality uses the assumption that $\mathcal{K}$ has diameter D . Using $\nabla_t = (\mathbf{x}_t^\top \mathbf{a}_t - b_t) \mathbf{a}_t$, we can rewrite the gradient term as $\nabla_t^\top A_t^{-1} \nabla_t = 2\ell_t(\mathbf{x}_t)(\mathbf{a}_t^\top A_t^{-1} \mathbf{a}_t)$. By our assumption that the losses are bounded by C , we have $\ell_t(\mathbf{x}_t) \leq C$, and thus

$$\sum_{t=1}^T \nabla_t^\top A_t^{-1} \nabla_t \leq 2C \sum_{t=1}^T \mathbf{a}_t^\top A_t^{-1} \mathbf{a}_t.$$

By the matrix determinant lemma, $A_t = A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top$, and the vector harmonic series lemma from [Haz16] Theorem 4.5, we have

$$\sum_t \mathbf{a}_t^\top A_t^{-1} \mathbf{a}_t = \sum_t A_t^{-1} \bullet (A_t - A_{t-1}) \leq \log \frac{\det(A_T)}{\det(A_0)}.$$

Thus, the regret is bounded by

$$\text{regret}_T(\mathbf{x}^*) \leq \frac{\lambda}{2} D^2 + C \log \frac{\det(A_T)}{\det(A_0)}.$$

To explicitly bound the log-determinant ratio, we bound the determinant of A_T using its trace. Since $A_T = \lambda I_d + \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top$, the trace is linearly bounded:

$$\text{Tr}(A_T) = \text{Tr}(\lambda I_d) + \sum_{t=1}^T \|\mathbf{a}_t\|_2^2 \leq \lambda d + TR^2.$$

By the AM-GM inequality, the determinant of a positive definite matrix is bounded by the average of its eigenvalues raised to the dimension d , which gives:

$$\det(A_T) \leq \left(\frac{\text{Tr}(A_T)}{d} \right)^d \leq \left(\frac{\lambda d + TR^2}{d} \right)^d \leq (\lambda + TR^2)^d.$$

Since $A_0 = \lambda I_d$, we have $\det(A_0) = \lambda^d$. Taking the logarithm of the ratio yields:

$$\log \frac{\det(A_T)}{\det(A_0)} \leq \log \frac{(\lambda + TR^2)^d}{\lambda^d} = d \log \left(1 + \frac{TR^2}{\lambda} \right).$$

Substituting this upper bound into our regret inequality yields the final stated bound. $\square$

A.4 The VAW Algorithm for Online Least Squares

We explicitly compare this approach to the well-known Vovk-Azoury-Warmuth (VAW) forecaster [Vov01, AW01] to highlight two critical distinctions. First, ONS operates strictly within the standard Online Convex Optimization (OCO) protocol, meaning no feature vector is revealed to the learner prior to making a decision. Second, ONS gracefully handles arbitrary constrained convex sets $\mathcal{K}$, whereas VAW is natively an unconstrained algorithm.

The projection step $\Pi_{\mathcal{K}}^{A_t}$ in Algorithm 11 plays a crucial mathematical role in keeping the algorithm stable within the strict OCO protocol. To see why, consider the *unprojected* version of the algorithm where $\mathcal{K} = \mathbb{R}^d$. The parameter update simplifies to:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - A_t^{-1} \mathbf{a}_t (\mathbf{x}_t^\top \mathbf{a}_t - b_t).$$

Multiplying both sides by $A_t = A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top$ gives:

$$A_t \mathbf{x}_{t+1} = (A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top) \mathbf{x}_t - \mathbf{a}_t \mathbf{a}_t^\top \mathbf{x}_t + b_t \mathbf{a}_t = A_{t-1} \mathbf{x}_t + b_t \mathbf{a}_t.$$

By induction (and assuming $A_0 \mathbf{x}_1 = 0$), we obtain:

$$\mathbf{x}_{t+1} = A_t^{-1} \sum_{s=1}^t b_s \mathbf{a}_s.$$

The vulnerability of unprojected ONS is that, because the decision $\mathbf{x}_t$ must be made *before* seeing $\mathbf{a}_t$, the resulting prediction $\mathbf{x}_t^\top \mathbf{a}_t$ can be overly aggressive.

A.4.1 The VAW Forecaster

The **Vovk-Azoury-Warmuth (VAW)** forecaster achieves logarithmic regret *without* requiring projections, but it must relax the strict OCO protocol

Algorithm 12 Vovk-Azoury-Warmuth (VAW) Forecaster

-
- 1: Input: regularization $\lambda > 0$
 - 2: Initialize $A_0 = \lambda I_d$, target vector $\mathbf{v}_0 = \mathbf{0} \in \mathbb{R}^d$
 - 3: **for** $t = 1$ to T **do**
 - 4: Observe feature vector $\mathbf{a}_t$
 - 5: Update curvature matrix: $A_t = A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top$
 - 6: Predict $\hat{b}_t = \mathbf{a}_t^\top A_t^{-1} \mathbf{v}_{t-1}$
 - 7: Observe true label b_t and suffer loss $\ell_t(\hat{b}_t) = \frac{1}{2}(\hat{b}_t - b_t)^2$
 - 8: Update target vector: $\mathbf{v}_t = \mathbf{v}_{t-1} + b_t \mathbf{a}_t$
 - 9: **end for**
-

to do so. In the VAW framework, the unlabelled feature $\mathbf{a}_t$ is revealed *before* the learner predicts.

Because $\mathbf{a}_t$ is known in advance, VAW exploits this by updating the precision matrix with the current feature before committing to a prediction. By the Sherman-Morrison formula, $A_t^{-1} \mathbf{a}_t = \frac{A_{t-1}^{-1} \mathbf{a}_t}{1 + \mathbf{a}_t^\top A_{t-1}^{-1} \mathbf{a}_t}$. Consequently, the VAW prediction can be written in terms of the unprojected ONS prediction (where $\mathbf{x}_t = A_{t-1}^{-1} \mathbf{v}_{t-1}$):

$$\hat{b}_t^{\text{VAW}} = \frac{\mathbf{x}_t^\top \mathbf{a}_t}{1 + \mathbf{a}_t^\top A_{t-1}^{-1} \mathbf{a}_t}.$$

This demonstrates that VAW fundamentally *shrinks* the standard unprojected Ridge Regression prediction by the factor $(1 + \mathbf{a}_t^\top A_{t-1}^{-1} \mathbf{a}_t)$. Whenever the model is highly uncertain in the direction of $\mathbf{a}_t$, the denominator grows and shrinks the prediction.

This implicit regularization automatically bounds the loss relative to the true labels, entirely bypassing the need for worst-case domain or magnitude bounds.

Theorem A.2 (Regret of the VAW Forecaster). *For Algorithm 12, assume the true labels are bounded such that $|b_t| \leq Y$ for all $t = 1, \dots, T$. For any benchmark $\mathbf{x}^* \in \mathbb{R}^d$, the regret is bounded by:*

$$\sum_{t=1}^T \frac{1}{2}(\hat{b}_t - b_t)^2 - \sum_{t=1}^T \frac{1}{2}(\mathbf{x}^{*\top} \mathbf{a}_t - b_t)^2 \leq \frac{\lambda}{2} \|\mathbf{x}^*\|_2^2 + \frac{Y^2}{2} \log \det \left(I_d + \frac{1}{\lambda} \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top \right).$$

In particular, if $\|\mathbf{a}_t\|_2 \leq R$, and setting $\lambda = 1/\|\mathbf{x}^\|^2$, with $d, Y, T, R > 2$ we get:*

$$\text{regret}_T(\mathbf{x}^*) \leq \frac{\lambda}{2} \|\mathbf{x}^*\|_2^2 + \frac{dY^2}{2} \log \left(1 + \frac{TR^2}{\lambda} \right) \leq 4dY^2 \log(TR\|\mathbf{x}^*\|).$$

Comparing Theorem A.1 to Theorem A.2 highlights a distinct advantage of VAW: the regret bound depends on Y^2 , which is determined strictly by the maximum magnitude of the observed labels. In contrast, the ONS bound depends on C , which scales with both the diameter of the decision set $\mathcal{K}$ and the feature norm R . However, this elegant shrinkage in VAW relies on the domain being unconstrained ($\mathbb{R}^d$) and requires violating the strict zero-lookahead OCO timeline, making projected ONS the necessary algorithm when dealing with strict constraints or true zero-lookahead settings.

A.4.2 VAW with Known Additive Offsets

We now consider a shifted prediction setting in which, at each round t , the learner observes both the feature vector $\mathbf{a}_t \in \mathbb{R}^d$ and a known scalar offset $c_t \in \mathbb{R}$ before making a prediction. The learner then predicts $\hat{z}_t \in \mathbb{R}$, after which the environment reveals the true label $z_t \in \mathbb{R}$, and the learner suffers square loss

$$\ell_t(\hat{z}_t) = \frac{1}{2}(\hat{z}_t - z_t)^2.$$

We compare against benchmarks of the form $\mathbf{x}^\star \top \mathbf{a}_t + c_t$, i.e. predictors that are allowed to use the same known offset:

$$\text{regret}_T^{\text{off}}(\mathbf{x}^\star) = \sum_{t=1}^T \frac{1}{2}(\hat{z}_t - z_t)^2 - \sum_{t=1}^T \frac{1}{2}(\mathbf{x}^\star \top \mathbf{a}_t + c_t - z_t)^2.$$

The key point is that, since c_t is known prior to prediction, one can simply subtract it from the labels and run the usual VAW forecaster on the centered targets.

Theorem A.3 (VAW regret with known additive offsets). *For Algorithm 13, assume the centered labels are bounded:*

$$|z_t - c_t| \leq Y \quad \text{for all } t = 1, \dots, T.$$

Then for any benchmark $\mathbf{x}^\star \in \mathbb{R}^d$,

$$\sum_{t=1}^T \frac{1}{2}(\hat{z}_t - z_t)^2 - \sum_{t=1}^T \frac{1}{2}(\mathbf{x}^\star \top \mathbf{a}_t + c_t - z_t)^2 \leq \frac{\lambda}{2} \|\mathbf{x}^\star\|_2^2 + \frac{Y^2}{2} \log \det \left(I_d + \frac{1}{\lambda} \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top \right).$$

In particular, the regret bound depends only on the centered labels $z_t - c_t$, and has no dependence on the magnitude of the offsets c_t .

If furthermore $\|\mathbf{a}_t\|_2 \leq R$ for all t , then

$$\sum_{t=1}^T \frac{1}{2}(\hat{z}_t - z_t)^2 - \sum_{t=1}^T \frac{1}{2}(\mathbf{x}^\star \top \mathbf{a}_t + c_t - z_t)^2 \leq \frac{\lambda}{2} \|\mathbf{x}^\star\|_2^2 + \frac{dY^2}{2} \log \left(1 + \frac{TR^2}{\lambda} \right).$$

Proof. Define the centered labels and centered predictions by

$$y_t := z_t - c_t, \quad \hat{y}_t := \hat{z}_t - c_t.$$

By construction of Algorithm 13, we have

$$\hat{y}_t = \mathbf{a}_t^\top A_t^{-1} \mathbf{v}_{t-1}, \quad \mathbf{v}_t = \mathbf{v}_{t-1} + y_t \mathbf{a}_t.$$

Therefore, when written in terms of $(\mathbf{a}_t, y_t)$, Algorithm 13 is exactly the standard VAW forecaster of Algorithm 12 applied to the centered labels y_t .

Since $|y_t| = |z_t - c_t| \leq Y$, Theorem A.2 implies that for every $\mathbf{x}^* \in \mathbb{R}^d$,

$$\sum_{t=1}^T \frac{1}{2} (\hat{y}_t - y_t)^2 - \sum_{t=1}^T \frac{1}{2} (\mathbf{x}^{*\top} \mathbf{a}_t - y_t)^2 \leq \frac{\lambda}{2} \|\mathbf{x}^*\|_2^2 + \frac{Y^2}{2} \log \det \left(I_d + \frac{1}{\lambda} \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top \right).$$

It remains to relate this centered regret to the original shifted problem. By definition of y_t and $\hat{y}_t$,

$$\hat{z}_t = \hat{y}_t + c_t, \quad z_t = y_t + c_t.$$

Hence the square loss is exactly translation invariant:

$$\frac{1}{2} (\hat{z}_t - z_t)^2 = \frac{1}{2} ((\hat{y}_t + c_t) - (y_t + c_t))^2 = \frac{1}{2} (\hat{y}_t - y_t)^2.$$

Likewise, for any benchmark $\mathbf{x}^*$,

$$\frac{1}{2} (\mathbf{x}^{*\top} \mathbf{a}_t + c_t - z_t)^2 = \frac{1}{2} (\mathbf{x}^{*\top} \mathbf{a}_t + c_t - (y_t + c_t))^2 = \frac{1}{2} (\mathbf{x}^{*\top} \mathbf{a}_t - y_t)^2.$$

Substituting these identities into the previous bound yields

$$\sum_{t=1}^T \frac{1}{2} (\hat{z}_t - z_t)^2 - \sum_{t=1}^T \frac{1}{2} (\mathbf{x}^{*\top} \mathbf{a}_t + c_t - z_t)^2 \leq \frac{\lambda}{2} \|\mathbf{x}^*\|_2^2 + \frac{Y^2}{2} \log \det \left(I_d + \frac{1}{\lambda} \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top \right),$$

which proves the first claim.

The simplified bound under $\|\mathbf{a}_t\|_2 \leq R$ follows exactly as in Theorem A.2, by bounding the log-determinant term via

$$\log \det \left(I_d + \frac{1}{\lambda} \sum_{t=1}^T \mathbf{a}_t \mathbf{a}_t^\top \right) \leq d \log \left(1 + \frac{TR^2}{\lambda} \right).$$

Thus the regret bound is completely independent of the magnitude of the offsets c_t ; only the centered labels $z_t - c_t$ matter. $\square$

Algorithm 13 VAW Forecaster with Known Additive Offsets

- 1: Input: regularization $\lambda > 0$
- 2: Initialize $A_0 = \lambda I_d$, target vector $\mathbf{v}_0 = \mathbf{0} \in \mathbb{R}^d$
- 3: **for** $t = 1$ to T **do**
- 4: Observe feature vector $\mathbf{a}_t$ and known offset c_t
- 5: Update curvature matrix:

$$A_t = A_{t-1} + \mathbf{a}_t \mathbf{a}_t^\top$$

- 6: Predict the centered label:

$$\hat{y}_t = \mathbf{a}_t^\top A_t^{-1} \mathbf{v}_{t-1}$$

- 7: Output the shifted prediction:

$$\hat{z}_t = \hat{y}_t + c_t$$

- 8: Observe true label z_t and suffer loss

$$\ell_t(\hat{z}_t) = \frac{1}{2}(\hat{z}_t - z_t)^2$$

- 9: Update target vector using the centered label:

$$\mathbf{v}_t = \mathbf{v}_{t-1} + (z_t - c_t) \mathbf{a}_t$$

- 10: **end for**

Bibliography

- [ABH⁺19] Naman Agarwal, Brian Bullins, Elad Hazan, Sham Kakade, and Karan Singh. Online control with adversarial disturbances. In *International Conference on Machine Learning*, pages 111–119. PMLR, 2019.
- [ACD⁺24] Naman Agarwal, Xinyi Chen, Evan Dogariu, Vlad Feinberg, Daniel Suo, Peter Bartlett, and Elad Hazan. Futurefill: fast generation from convolutional sequence models. *arXiv preprint arXiv:2410.03766*, 2024.
- [Ada20] David Adam. Special report: The simulations driving the world’s response to covid-19. *Nature*, 580(7802):316–319, 2020.
- [ADLM19] James Anderson, John C Doyle, Steven H Low, and Nikolai Matni. System level synthesis. *Annual Reviews in Control*, 47:364–393, 2019.
- [ADT12] Raman Arora, Ofer Dekel, and Ambuj Tewari. Online bandit learning against an adaptive adversary: from regret to policy regret. *arXiv preprint arXiv:1206.6400*, 2012.
- [ÅH95] Karl Johan Åström and Tore Hägglund. *PID controllers: Theory, design, and tuning*, volume 2. Instrument society of America Research Triangle Park, NC, 1995.
- [AHL⁺18] Sanjeev Arora, Elad Hazan, Holden Lee, Karan Singh, Cyril Zhang, and Yi Zhang. Towards provable control for unknown linear dynamical systems. 2018.
- [Ahm12] Amir Ali Ahmadi. On the difficulty of deciding asymptotic stability of cubic homogeneous vector fields. In *2012 American Control Conference (ACC)*, pages 3334–3339. IEEE, 2012.

- [AHM13] Oren Anava, Elad Hazan, and Shie Mannor. Online convex optimization against adversaries with memory and application to statistical arbitrage. *arXiv preprint arXiv:1302.6937*, 2013.
- [Ahm16] Amir Ali Ahmadi. ORF 363: Computing and Optimization. <http://aaa.princeton.edu/orf363>, 2016. [Online; accessed August 1, 2022].
- [AHMS21] Naman Agarwal, Elad Hazan, Anirudha Majumdar, and Karan Singh. A regret minimization approach to iterative learning control. In *International Conference on Machine Learning*, pages 100–109. PMLR, 2021.
- [AHS19] Naman Agarwal, Elad Hazan, and Karan Singh. Logarithmic regret for online control. In *Advances in Neural Information Processing Systems*, pages 10175–10184, 2019.
- [AJKS19] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Lectures on the theory of reinforcement learning, 2019.
- [AM79] Brian D. O. Anderson and John B. Moore. *Optimal filtering*. Dover Publications, 1979.
- [AM91] BDO Anderson and JB Moore. Kalman filtering: Whence, what and whither? *Mathematical system theory: the influence of RE Kalman*, pages 41–54, 1991.
- [Ant05] Athanasios C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.
- [ASCH23] Naman Agarwal, Daniel Suo, Xinyi Chen, and Elad Hazan. Spectral state space models. *arXiv preprint arXiv:2312.06837*, 2023.
- [AW01] Katy S Azoury and Manfred K Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.
- [AYBK14] Yasin Abbasi-Yadkori, Peter Bartlett, and Varun Kanade. Tracking adversarial targets. In *International Conference on Machine Learning*, pages 369–377. PMLR, 2014.
- [AYS11] Yasin Abbasi-Yadkori and Csaba Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In *Proceedings*

- of the 24th Annual Conference on Learning Theory, pages 1–26, 2011.
- [BB08] Tamer Başar and Pierre Bernhard. *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer Science & Business Media, 2008.
- [BBDH25] Anand Brahmabhatt, Gon Buzaglo, Sofia Druchyna, and Elad Hazan. A new approach to controlling linear dynamical systems, 2025.
- [BBKK22] Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. Modern koopman theory for dynamical systems. *SIAM Review*, 64(2):229–340, 2022.
- [BBPK16] Steven L. Brunton, Bingni W. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLOS ONE*, 11(2):e0150171, 2016.
- [Ben93] S. Bennett. Development of the pid controller. *IEEE Control Systems Magazine*, 13(6):58–62, 1993.
- [Ber07] Dimitri P Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, 2007.
- [Ber19] Dimitri Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.
- [BLMY23a] Ainesh Bakshi, Allen Liu, Ankur Moitra, and Morris Yau. A new approach to learning linear dynamical systems. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 335–348, 2023.
- [BLMY23b] Ainesh Bakshi, Allen Liu, Ankur Moitra, and Morris Yau. Tensor decompositions meet control theory: learning general mixtures of linear dynamical systems. In *International Conference on Machine Learning*, pages 1549–1563. PMLR, 2023.
- [BT97] Vincent Blondel and John N Tsitsiklis. Np-hardness of some linear control design problems. *SIAM journal on control and optimization*, 35(6):2118–2127, 1997.

- [BT99] Vincent D Blondel and John N Tsitsiklis. Three problems on the decidability and complexity of stability. In *Open problems in mathematical systems and control theory*, pages 45–51. Springer, 1999.
- [BT00] Vincent D Blondel and John N Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, 2000.
- [BT17] Bernhard Beckermann and Alex Townsend. On the singular values of matrices with displacement structure. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1227–1248, 2017.
- [BW22] Dheeraj Baby and Yu-Xiang Wang. Optimal dynamic regret in lqr control. *Advances in Neural Information Processing Systems*, 35:24879–24892, 2022.
- [BY08] Mark Braverman and Michael Yampolsky. *Computability of Julia sets*. Springer, 2008.
- [CBL06] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.
- [CCK20] Asaf Cassel, Alon Cohen, and Tomer Koren. Logarithmic regret for learning linear quadratic regulators efficiently. In *International Conference on Machine Learning*, pages 1328–1337. PMLR, 2020.
- [CCK22] Asaf Cassel, Alon Cohen, and Tomer Koren. Rate-optimal online convex optimization in adaptive linear control. *arXiv preprint arXiv:2206.01426*, 2022.
- [CH21] Xinyi Chen and Elad Hazan. Black-box control for linear dynamical systems. In *Conference on Learning Theory*, pages 1114–1143. PMLR, 2021.
- [CHK⁺18] Alon Cohen, Avinatan Hasidim, Tomer Koren, Nevena Lazic, Yishay Mansour, and Kunal Talwar. Online linear quadratic control. In *International Conference on Machine Learning*, pages 1029–1038, 2018.
- [CK20] Asaf Cassel and Tomer Koren. Bandit linear control. *Advances in Neural Information Processing Systems*, 33:8872–8882, 2020.

- [CKM19] Alon Cohen, Tomer Koren, and Yishay Mansour. Learning linear-quadratic regulators efficiently with only $\sqrt{T}$ regret. In *International Conference on Machine Learning*, pages 1300–1309, 2019.
- [DBH25] Evan Dogariu, Anand Brahmbhatt, and Elad Hazan. Universal learning of nonlinear dynamics. *arXiv preprint arXiv:2508.11990*, 2025.
- [DKL⁺21] Simon Du, Sham Kakade, Jason Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear classes: A structural framework for provable generalization in rl. In *International Conference on Machine Learning*, pages 2826–2836. PMLR, 2021.
- [DMM⁺18] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. Regret bounds for robust adaptive control of the linear quadratic regulator. In *Advances in Neural Information Processing Systems*, pages 4188–4197, 2018.
- [DMM⁺20] Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, 20(4):633–679, 2020.
- [DMW⁺17] John C Doyle, Nikolai Matni, Yuh-Shyang Wang, James Anderson, and Steven Low. System level synthesis: A tutorial. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2856–2867. IEEE, 2017.
- [Doy78] John C Doyle. Guaranteed margins for lqg regulators. *IEEE Transactions on automatic Control*, 23(4):756–757, 1978.
- [EDKM04] Eyal Even-Dar, Sham M Kakade, and Yishay Mansour. Experts in a markov decision process. *Advances in neural information processing systems*, 17, 2004.
- [FCZI03] Enrique Fernández Cara and Enrique Zuazua Iriondo. Control theory: History, mathematical achievements and perspectives. *Boletín de la Sociedad Española de Matemática Aplicada*, 26, 79-140., 2003.

- [FGKM18] Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International Conference on Machine Learning*, pages 1467–1476, 2018.
- [Fie97] Claude-Nicolas Fiechter. Pac adaptive control of linear systems. In *Proceedings of the tenth annual conference on Computational learning theory*, pages 72–80, 1997.
- [Fro99] Gary Froyland. Using ulam’s method to calculate entropy and other dynamical invariants. *Nonlinearity*, 12(1):79–101, 1999.
- [Fro01] Gary Froyland. Extracting dynamical behavior via markov models. In Alistair I. Mees, editor, *Nonlinear Dynamics and Statistics*, pages 281–321. Birkhäuser Boston, Boston, MA, 2001.
- [FS20] Dylan J. Foster and Max Simchowitz. Logarithmic regret for adversarial online control, 2020.
- [FSR20] Dylan J. Foster, Tuhin Sarkar, and Alexander Rakhlin. Learning nonlinear dynamical systems from a single trajectory. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, pages 851–861. PMLR, 2020.
- [FZPK19] Luca Furieri, Yang Zheng, Antonis Papachristodoulou, and Maryam Kamgarpour. An input–output parametrization of stabilizing controllers: Amidst youla and system level synthesis. *IEEE Control Systems Letters*, 3(4):1014–1019, 2019.
- [GASH22] Gautam Goel, Naman Agarwal, Karan Singh, and Elad Hazan. Best of both worlds in online control: Competitive ratio and policy regret. *arXiv preprint arXiv:2211.11219*, 2022.
- [GGR22] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *International Conference on Learning Representations (ICLR)*, 2022.
- [GH21] Gautam Goel and Babak Hassibi. Competitive control. *arXiv preprint arXiv:2107.13657*, 2021.
- [GHH20] Paula Gradu, John Hallman, and Elad Hazan. Non-stochastic control with bandit feedback. *Advances in Neural Information Processing Systems*, 33:10764–10774, 2020.

- [GHL⁺24] Noah Golowich, Elad Hazan, Zhou Lu, Dhruv Rohatgi, and Y Jennifer Sun. Online control in population dynamics. *arXiv preprint arXiv:2406.01799*, 2024.
- [GHM20] Paula Gradu, Elad Hazan, and Edgar Minasyan. Adaptive regret for control of time-varying dynamics. *arXiv preprint arXiv:2007.04393*, 2020.
- [GLS⁺20] Udaya Ghai, Holden Lee, Karan Singh, Cyril Zhang, and Yi Zhang. No-regret prediction in marginally stable systems. In *Conference on Learning Theory*, pages 1714–1757. PMLR, 2020.
- [GLSW19] Gautam Goel, Yiheng Lin, Haoyuan Sun, and Adam Wierman. Beyond online balanced descent: An optimal algorithm for smoothed online optimization. *Advances in Neural Information Processing Systems*, 32, 2019.
- [GSS93] Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE proceedings F (radar and signal processing)*, volume 140, pages 107–113. IET, 1993.
- [GW19] Gautam Goel and Adam Wierman. An online algorithm for smoothed regression and lqr control. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2504–2513. PMLR, 2019.
- [Haz16] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.
- [Haz19] Elad Hazan. Lecture notes: Optimization for machine learning, 2019.
- [Hes18] Joao P Hespanha. *Linear systems theory*. Princeton university press, 2018.
- [HK66] BL HO and Rudolf E Kálmán. Effective construction of linear state-variable models from input/output functions. *at-Automatisierungstechnik*, 14(1-12):545–548, 1966.
- [HKS20] Elad Hazan, Sham Kakade, and Karan Singh. The nonstochastic control problem. In *Algorithmic Learning Theory*, pages 408–421. PMLR, 2020.

- [HLS⁺18] Elad Hazan, Holden Lee, Karan Singh, Cyril Zhang, and Yi Zhang. Spectral filtering for general linear dynamical systems. In *Advances in Neural Information Processing Systems*, pages 4634–4643, 2018.
- [HM26] Elad Hazan and Annie Marsden. Spectral filtering for learning quantum dynamics, 2026.
- [HMR18] Moritz Hardt, Tengyu Ma, and Benjamin Recht. Gradient descent learns linear dynamical systems. *Journal of Machine Learning Research*, 19(29):1–44, 2018.
- [HS21] Elad Hazan and Karan Singh. ICML tutorial: Online and nonstochastic control. In *International Conference on Machine Learning*, 2021.
- [HSK96] Babak Hassibi, Ali H Sayed, and Thomas Kailath. Linear estimation in krein spaces. ii. applications. *IEEE Transactions on Automatic Control*, 41(1):34–49, 1996.
- [HSZ17] Elad Hazan, Karan Singh, and Cyril Zhang. Learning linear dynamical systems via spectral filtering. In *Advances in Neural Information Processing Systems*, pages 6702–6712, 2017.
- [IJR12] Morteza Ibrahimi, Adel Javanmard, and Benjamin Roy. Efficient reinforcement learning for high dimensional linear quadratic systems. *Advances in Neural Information Processing Systems*, 25, 2012.
- [JKA⁺17] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low bellman rank are pac-learnable. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2017.
- [JU04] Simon J Julier and Jeffrey K Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.
- [JYWJ20] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pages 2137–2143. PMLR, 2020.

- [Kal60] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82.1:35–45, 1960.
- [Kha15] Hassan K Khalil. *Nonlinear control*, volume 406. Pearson New York, 2015.
- [KKBK20] Mason Kamb, Eurika Kaiser, Steven L. Brunton, and J. Nathan Kutz. Time-delay observables for koopman: Theory and applications. *SIAM Journal on Applied Dynamical Systems*, 19(2):886–917, 2020.
- [KKL⁺20] Sham Kakade, Akshay Krishnamurthy, Kendall Lowrey, Motoya Ohnishi, and Wen Sun. Information theoretic regret bounds for online nonlinear control. *Advances in Neural Information Processing Systems*, 33:15312–15325, 2020.
- [KM18] Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018.
- [KMTM19] Mark Kozdoba, Jakub Marecek, Tigran Tchrakian, and Shie Mannor. On-line learning of linear dynamical systems: Exponential forgetting in kalman filters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4098–4105, 2019.
- [KNP⁺20] Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020.
- [Koo31] Bernard O Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.
- [LAHA20a] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Logarithmic regret bound in partially observable linear dynamical systems, 2020.
- [LAHA20b] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Regret bound of adaptive control in linear quadratic gaussian (lqg) systems, 2020.

- [LAHA20c] Sahin Lale, Kamyar Azizzadenesheli, Babak Hassibi, and Anima Anandkumar. Regret minimization in partially observable linear quadratic control, 2020.
- [LDL21] Yingying Li, Subhro Das, and Na Li. Online optimal control with affine constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8527–8537, 2021.
- [Li05] Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 300–306. IEEE, 2005.
- [Lju10] Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.
- [LK13] Sergey Levine and Vladlen Koltun. Guided policy search. In *International conference on machine learning*, pages 1–9. PMLR, 2013.
- [LND⁺24] Y Isabel Liu, Windsor Nguyen, Yagiz Devre, Evan Dogariu, Anirudha Majumdar, and Elad Hazan. Flash stu: Fast spectral transform units. *arXiv preprint arXiv:2409.10489*, 2024.
- [Lor63] Edward N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, mar 1963.
- [Lue64] David Luenberger. Observing the state of a linear system. *IEEE Transactions on Military Electronics*, 8(2):74–80, 1964.
- [Lue71] David Luenberger. An introduction to observers. *IEEE Transactions on Automatic Control*, 16(6):596–602, 1971.
- [Lya92] Aleksandr Mikhailovich Lyapunov. The general problem of the stability of motion. *International journal of control*, 55(3):531–534, 1992.
- [Max68] James Clerk Maxwell. I. on governors. *Proceedings of the Royal Society of London*, 16:270–283, 1868.
- [MDA⁺24] Annie Marsden, Evan Dogariu, Naman Agarwal, Xinyi Chen, Daniel Suo, and Elad Hazan. Provable length generalization in sequence prediction via spectral filtering. *arXiv preprint arXiv:2411.01035*, 2024.

- [Mez05] Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1–3):309–325, 2005.
- [Mez13] Igor Mezić. Analysis of fluid flows via spectral properties of the koopman operator. *Annual Review of Fluid Mechanics*, 45:357–378, 2013.
- [MGSH21] Edgar Minasyan, Paula Gradu, Max Simchowitz, and Elad Hazan. Online control of unknown time-varying dynamical systems. *Advances in Neural Information Processing Systems*, 34:15934–15945, 2021.
- [MH25] Annie Marsden and Elad Hazan. Dimension-free regret for learning asymmetric linear dynamical systems. *arXiv preprint arXiv:2502.06545*, 2025.
- [MMS20] Alexandre Mauroy, Igor Mezić, and Yoshihiko Susuki, editors. *The Koopman Operator in Systems and Control: Concepts, Methodologies, and Applications*, volume 484 of *Lecture Notes in Control and Information Sciences*. Springer, 2020.
- [MTR19] Horia Mania, Stephen Tu, and Benjamin Recht. Certainty equivalence is efficient for linear quadratic control. In *Advances in Neural Information Processing Systems*, pages 10154–10164, 2019.
- [OO19] Samet Oymak and Necmiye Ozay. Non-asymptotic identification of lti systems from a single trajectory. In *2019 American control conference (ACC)*, pages 5655–5661. IEEE, 2019.
- [OR21] Samuel E. Otto and Clarence W. Rowley. Koopman operators for estimation and control of dynamical systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:59–87, 2021.
- [PBK18] Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz. Generalizing koopman theory to allow for inputs and control. *SIAM Journal on Applied Dynamical Systems*, 17(1):909–930, 2018.
- [Pel03] Vladimir V. Peller. *Hankel Operators and Their Applications*. Springer, 2003.

- [PH20] Orestis Plevrakis and Elad Hazan. Geometric exploration for online control. *Advances in Neural Information Processing Systems*, 33:7637–7647, 2020.
- [RN02] P R Russel and S A Norvig. Artificial intelligence: A modern approach. *Prentice Hall Upper Saddle River, NJ, USA: Rani, M., Nayak, R., & Vyas, OP (2015). An ontology-based adaptive personalized e-learning system, assisted by software agents on cloud storage. Knowledge-Based Systems*, 90:33–48, 2002.
- [Roc87] R Tyrell Rockafellar. Linear-quadratic programming and optimal control. *SIAM Journal on Control and Optimization*, 25(3):781–814, 1987.
- [SB18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.
- [SBB90] Shankar Sastry, Marc Bodson, and James F Bartram. Adaptive control: stability, convergence, and robustness, 1990.
- [SBR19] Max Simchowitz, Ross Boczar, and Benjamin Recht. Learning linear dynamical systems with semi-parametric least squares. In *Conference on Learning Theory*, pages 2714–2802. PMLR, 2019.
- [Sch10] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.
- [SF20] Max Simchowitz and Dylan Foster. Naive exploration is optimal for online LQR. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8937–8948. PMLR, 13–18 Jul 2020.
- [Sil15] David Silver. Lecture notes on reinforcement learning. Lecture Notes, 2015.
- [Sim20] Max Simchowitz. Making non-stochastic control (almost) as easy as stochastic. *Advances in Neural Information Processing Systems*, 33:18318–18329, 2020.
- [Sin22] Karan Singh. *The Nonstochastic Control Problem*. PhD thesis, Princeton University, 2022.

- [SL91] Jean-Jacques E Slotine and Weiping Li. *Applied nonlinear control*. Prentice Hall, 1991.
- [SL24] Y Jennifer Sun and Zhou Lu. Tight rates for bandit control beyond quadratics. *arXiv preprint arXiv:2410.00993*, 2024.
- [SLC⁺20] Guanya Shi, Yiheng Lin, Soon-Jo Chung, Yisong Yue, and Adam Wierman. Online optimization with memory and competitive control. *Advances in Neural Information Processing Systems*, 33:20636–20647, 2020.
- [SMT⁺18] Max Simchowitz, Horia Mania, Stephen Tu, Michael I Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. In *Conference On Learning Theory*, pages 439–473. PMLR, 2018.
- [SNH23] Y Jennifer Sun, Stephen Newman, and Elad Hazan. Optimal rates for bandit nonstochastic control. *Advances in Neural Information Processing Systems*, 36:21908–21919, 2023.
- [SO22] Yahya Sattar and Samet Oymak. Non-asymptotic and accurate learning of nonlinear dynamical systems. *Journal of Machine Learning Research*, 23(140):1–49, 2022.
- [SR19] Tuhin Sarkar and Alexander Rakhlin. Near optimal finite time identification of arbitrary linear dynamical systems. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5610–5618. PMLR, 09–15 Jun 2019.
- [SSH20] Max Simchowitz, Karan Singh, and Elad Hazan. Improper learning for non-stochastic control. In *Conference on Learning Theory*, pages 3320–3436. PMLR, 2020.
- [SSNH24] Arun Suggala, Y Jennifer Sun, Praneeth Netrapalli, and Elad Hazan. Second order methods for bandit optimization and control. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 4691–4763. PMLR, 2024.
- [Ste94] Robert F Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

- [Str14] Steven H. Strogatz. *Nonlinear Dynamics and Chaos*. Taylor & Francis Inc, 2014.
- [SWN11] Thomas B. Schön, Adrian Wills, and Brett Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.
- [SZG⁺21] Daniel Suo, Cyril Zhang, Paula Gradu, Udaya Ghai, Xinyi Chen, Edgar Minasyan, Naman Agarwal, Karan Singh, Julianne LaChance, Tom Zajdel, et al. Machine learning for mechanical ventilation control. *arXiv preprint arXiv:2102.06779*, 2021.
- [TE05] Lloyd N Trefethen and Mark Embree. *Spectra and pseudospectra: the behavior of nonnormal matrices and operators*. Princeton University Press, 2005.
- [Ted20] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832)*. 2020.
- [TKY17] Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems 30*, pages 1130–1140, 2017.
- [TRL⁺14] Jonathan H Tu, Clarence W Rowley, Dirk M Luchtenburg, Steven L Brunton, and J Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.
- [Tro12] Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.
- [Tu18] Stephen Tu. H-infinity optimal control via dynamic games, 2018.
- [Tuc99] Warwick Tucker. The lorenz attractor exists. *Comptes Rendus de l’Académie des Sciences-Series I-Mathematics*, 328(12):1197–1202, 1999.
- [UA21] Claudio Urrea and Rayko Agramonte. Kalman filter: historical overview and review of its use in robotics 60 years after its creation. *Journal of Sensors*, 2021(1):9674015, 2021.

- [Ula60] Stanislaw M. Ulam. *A Collection of Mathematical Problems*, volume 8 of *Interscience Tracts in Pure and Applied Mathematics*. Interscience Publishers, New York, 1960.
- [Vov01] Volodya Vovk. Competitive on-line statistics. *International Statistical Review*, 69(2):213–248, 2001.
- [Wie49] Norbert Wiener. *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*. MIT press Cambridge, MA, 1949.
- [WKR15] Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Non-linear Science*, 25:1307–1346, 2015.
- [Won67] W Wonham. On pole assignment in multi-input controllable linear systems. *IEEE Transactions on Automatic Control*, 12(6):660–665, 1967.
- [Won68] Walter Murray Wonham. On the separation theorem of stochastic control. *SIAM Journal on Control*, 6(2):312–326, 1968.
- [YJB76] Dante Youla, Hamid Jabr, and Jr Bongiorno. Modern wiener-hopf design of optimal controllers–part ii: The multivariable case. *IEEE Transactions on Automatic Control*, 21(3):319–338, 1976.
- [ZDG96a] Kemin Zhou, JC Doyle, and Keither Glover. Robust and optimal control. *Control Engineering Practice*, 4(8):1189–1190, 1996.
- [ZDG96b] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice-Hall, Inc., USA, 1996.
- [ZFP⁺20] Yang Zheng, Luca Furieri, Antonis Papachristodoulou, Na Li, and Maryam Kamgarpour. On the equivalence of youla, system-level, and input-output parameterizations. *IEEE Transactions on Automatic Control*, 66(1):413–420, 2020.
- [ZWZ22] Peng Zhao, Yu-Xiang Wang, and Zhi-Hua Zhou. Non-stationary online learning with memory and non-stochastic control. In *International Conference on Artificial Intelligence and Statistics*, pages 2101–2133. PMLR, 2022.

- [ZZLL23] Runyu Cathy Zhang, Yang Zheng, Weiyu Li, and Na Li. On the relationship of optimal state feedback and disturbance response controllers. *IFAC-PapersOnLine*, 56(2):7503–7508, 2023.