The Nonstochastic Control Problem

Karan Singh

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
by the Department of
Computer Science
Adviser: Elad Hazan

January 2022

Abstract

Controlling a dynamical system is a fundamental problem, with wide-reaching applications in autonomous systems, engineering, and sciences. This thesis outlines algorithms and their analyses for feedback-driven learning that make progress on several fronts: robustness, adaptivity, generality, and sample efficiency.

The focal point of this thesis is the introduction of the nonstochastic control framework, which builds an algorithmic, against the traditionally analytic, foundation for control theory. This segment provides robust algorithmic primitives for basic control subroutines. Concretely, it presents:

1. An efficient algorithm for controlling linear systems in the presence of nonstochastic perturbations with an instance-optimal regret guarantee. Beyond the realm of both robust and stochastic control, such a data-driven notion of optimality offers worst-case guarantees with a promise of exceptional performance on benign problem instances;

2. The first logarithmic regret result for the classical tracking problem, using optimization-based tools that go beyond dynamic programming based approaches;

3. Extensions of the above to the case of unknown system dynamics.

The second part of the thesis addresses the challenge of learning to act in non-stationary environments given a handful of trials, a version of the sim-to-real challenge. It provides a provable algorithm that negotiates this challenge using the algorithmic foundation delineated above, and demonstrates its efficacy over known approaches in non-linear settings.

The second part of the thesis concerns the generalization of the notion of feedback in interactive learning beyond scalar rewards. It provides an efficient reductions-based approach to tackling concave reward functionals, that may increasingly be found in underlying exploration subroutines. This is accomplished via a novel generalization of classical boosting and gradient boosting techniques to the online convex optimization setting.

Acknowledgements

I would like to begin by expressing my graditude to my adviser, Elad Hazan, for his constant support, his generous guidance both on academic and personal fronts, and for inculcating in me a sense of optimisim and wonder towards research. I hope that my stay at Princeton has instilled in me, even if in small amounts, his elegant taste in problems, and his persistent focus on simplicity in solutions.

I would also like to thank Sanjeev Arora, Sham Kakade and Ani Majumdar for their mentorship, and Karthik Narasimhan for agreeing to be on my dissertation committee.

I would like to express my thanks to my academic kith-kin and friends – Cyril, Naman, Brian, Yi, Holden, Xinyi, Udaya, Nataly, Edgar, Daniel, Paula, Max, Abby – for being great collaborators on numerous adventures and misadventures.

My parents have been a contant source of love, support and encouragement in an otherwise ever-changing universe. To them, I am eternally indebted.

Parts of this thesis were supported by Elad Hazan's NSF grants IIS-1523815 and CCF1704860, and the generous support of the Porter Ogden Jacobus Fellowship.

Contents

Chapter 1

Introduction

Machine learning is often defined as the design and study of algorithms whose performance improves with successive acquisition of experience [Mit97]. It is thus possible to classify machine learning problems depending on how one defines a measure of "performance" and the nature of "experience". The most prevalent and mature paradigm of machine learning is supervised learning, where "experience" is a stream of example feature-label pairs, "performance" is defined as the average cost of the algorithm's predictions against the true labels as revealed in the stream of examples. The theory of statistical learning [Vap99] lays down both sufficient and necessary conditions for learning with finite examples when each member of the stream of examples is sampled independently each time from a fixed distribution (i.i.d.) over feature-label pairs. The last of these assumptions (i.i.d.) is not often not realistic, e.g. the need for a pumpkin is not a constant, and one might be more inclined to buy one near Halloween. Online learning [H$^+$16, SS$^+$12] rose in response to such considerations of nonstationarity; specifically online learning makes no distributional assumptions, i.e. is nonstochastic, on the observed feature-label pair sequence. In practice too, in the past decade, aided by advances in computing and availability of large-scale training datasets, supervised learning algorithms, quite often neural network based, have driven successes in many domains; machine translation and image recognition are prominent examples.

Reinforcement learning (or control) deals with the design of automated agents that learn to act performatively in stateful environments. The central distinction between reinforcement learning and supervised learning is that in the former the agent's actions have an impact on the environment, thereby altering the future experiences it might encounter, i.e. current actions have future consequences. Mirroring the development of statistical learning, the prevailing way of describing such decision

making scenarios is to model the environment as a Markov decision process (MDP), where the Markov assumption on state evolution is an analogue of the (conditional) i.i.d. assumption from supervised learning. Similarly, the Markov assumption is often violated in practice; the lack in quantity or quality of sensors means that the current observation might not be rich enough to accurately predict (distributionally) the next state or the reward. Partially observed MDPs (POMDPs) which get rid of the Markov assumption on observations are known to be hard to scale beyond discrete state-action setting both in theory and practice, and retain the disadvantages of making a supposition of stochasticity.

At the broadest outlook, this thesis deals with the issues of nonstationarity and nonstochasticity in such feedback-driven learning systems, and proposes statistically and computationally efficient algorithmic solutions to either mitigate their adverse effects on performance, or, in some case, to opportunistically make use of them to even accelerate learning. The results here may be viewed as early steps in a developing a theory of (nonstochastic) online learning for stateful interaction-driven environments.

1.1 Overview of contributions

Stochasticity and stationarity, or lack thereof, occur in reactive environments in two forms: a (distributional or) Markov assumption on state evolution or dynamics, and a conditional independence assumption on the immediate feedback in terms of cost or reward. The absence of either of the previous assumptions renders the decision making problem a non-MDP. This thesis makes progress on both fronts.

1.1.1 The nonstochastic control problem

In the first part, we consider continuous control problems that are subject to an arbitrary perturbation sequence affecting the state evolution and an arbitrary cost function sequence: nonstochastic control problems. The traditional treatment of perturbations in control theory is through the analytic design of robust controllers, which minimize the cost of the controller (or agent) against the worst-case perturbation sequence. The cost of this robustness is that such controllers perform significantly sub-optimally on average-case instances. We propose the benchmark of regret, the difference between the cost of the agent and that of an omniscient policy that has complete foreknowledge of future costs and perturbations. A sublinear regret guarantee certifies instance-optimality (vs. worst-case optimality of robust controllers), since the regret guarantee holds against against the perturbation-

cost sequence actually observed. Such an instance-optimality guarantee implies that the agent's performance matches the performance of a stochastic-optimal controller on average-case instances, and that of a robust controller on worst-case instances, while interpolating the performance gracefully in between. In fact, on benign or predictable perturbation sequences, such an instance-optimal controller can perform significantly better than both the stochastic-optimal and the robust controller, therefore opportunistically making use of the ease of the instance.

Concretely, consider the following dynamical description. This may be seen as discrete-time analogues of controlled diffusion processes [Kry08].

$$x_{t+1} = f_t(x_t, u_t, w_t)$$

Here $x_t \in \mathbb{R}^m$ is the state, $u_t \in \mathbb{R}^n$ is the control input, $f_t(x, u, w)$ is a transition function, and w_t are perturbations or deviations from nominal dynamics. The agent or the controller must iteratively choose a control input $u_t \in \mathbb{R}^n$, and suffers a loss $c_t(x_t, u_t)$. The controller is thereafter presented some output in the form of a resultant information set $\mathcal{I}_t$. For example, the controller may observe the state x_t, and cost function c_t, but the transition function f might be unknown. A policy $\pi = (\pi_1, \dots \pi_t : \pi_t :: \mathcal{I}_t \to \mathbb{R}^n)$ is a mapping from observed information to control. We denote a set of policies by Π. We measure the performance of a control algorithm through the metric of policy regret: the difference between the aggregate cost of the controller and that of the best policy in hindsight from a certain class.

Definition 1.1.1 (Nonstochastic Control)**.** *A Nonstochastic Control problem instance is given by tuple* $(f : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m, \{w_t \in \mathbb{R}^m\}, \{c_t : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}\}, \Pi)$*, where for any sequence of controls* $u_1, ..., u_T$*, the states are produced as* $x_{t+1} = f(x_t, u_t, w_t)$*. The goal of the learner is to choose an adaptive sequence of controls to minimize regret against the policy class* Π*, defined as:*

$$\textit{Regret} = \sum_{t=1}^{T} c_t(x_t, u_t) - \min_{\pi \in \Pi} \sum_{t=1}^{T} c_t(x_t^\pi, u_t^\pi),$$

where (x_t^π, u_t^π) *is the state-control pair visited by the benchmark policy* $\pi \in \Pi$ *in consideration.*

In chapter 2, using tools from online convex optimization [H+16] and a novel controller parameterization, we design an algorithm that achieves a sublinear guarantee for linear control, thus delivering on the benefits promised in the previous paragraph. A crucial distinction here in contrast to traditional control theoretic approaches is that the resultant control law can not be pinned down analytically, and is instead the outcome of an optimization-based learning process. We therefore view

this is as a first step towards establishing an algorithmic foundation for control theory.

Subsequently, in chapter 3, we show that the tools developed for nonstochastic control are useful even in a stochastic setup. Concretely, we give the first logarithmic regret algorithm for the classical online tracking problem, a practically relevant setup considered since the early days of control theory [Kal60].

In chapter 4, we extend the results to more permissive settings, such as when the system matrices that parameterize the dynamics are unknown.

In chapter 5, we consider the task of adaptive episodic planning in settings where there is a mismatch between the nominal dynamics the learner has explicit access to and the true world dynamics that learner must operate in. This is a version of the sim-to-real problem. Here, we propose an algorithmic solution derived on nonstochastic principles that is competitive in an average cost sense against the best omniscient policy that has both an episode-agnostic and an episode-specific policy components. Such a guarantee combines inter- and intra-episodic learning objectives, promising simultaneously good performance on individual episodes, while retaining the ability to generalize knowledge and computation across episodes. We show the empirical efficacy of the proposal in both linear and non-linear control settings.

1.1.2 Boosting in reinforcement learning

The framework of MDPs in reinforcement learning relies on so-called scalar rewards, a real-valued function evaluated at the state-action pair the agent visits, to incentivize the desired agent behavior. In this part of the thesis, we consider generalizing the notion of feedback the agent may receive, while not compromising on the statistical and computational benefits (e.g. via bootstrapping values, dynamic programming) of the scalar reward model of interaction. In particular, we consider the case when the agent has to maximize a concave reward functional evaluated at the state distribution induced by the agent's action. This form of feedback is strictly more general than scalar rewards which are linear functions of the state distribution. The importance of such objectives lies in task-agnostic exploration settings, where the agent is incentivized to visit new parts of the state space which requires a notion of history to be built into the agent's reward model. The direct optimization of such temporally-extended global modes of feedback pose a challenge in that from the agent's perspective the reward model seems highly non-stationary. We deal with this challenge by casting the reward functional maximization problem into the framework of gradient boosting. To accomplish this translation, we first strengthen the guarantees for (agnostic) boosting present in the literature,

which may be of independent interest.

Boosting is a computational framework for compositional learning. There are two traditions to the theory of boosting. First: classical boosting, arising from theoretical CS, converts weak slightly-better-than-random learners to an accurate one, enhancing the accuracy. Originally designed for the binary classification setting, the literature on boosting was extended to multi-class, multi-label, and ranking-based settings (all examples of linear loss) with specialized constructions in each case. Second: gradient boosting, on the other hand, aggregates simple (but accurate) learners into a more expressive one; it guarantees competitiveness with the convex hull of the weak hypothesis class. The main contribution detailed in chapter 6 is an efficient algorithm that enhances the accuracy and expressivity of the learning process at the same time, while operating on general convex loss (vs. linear for classical boosting) and any convex decision set. This resultant excess risk (average regret) guarantee unifies and delivers on the twin objectives of classical boosting and gradient boosting. The reduction holds for both the (non-stochastic) online and statistical settings, and is amenable to bandit feedback.

Building on this result, in chapter 7, we give an efficient algorithm for maximizing concave reward functionals in a reinforcement learning setup, given access to a blackbox planning oracle for scalar reward functions (the mainstay in MDPs), and a density estimation oracle for the agent's state distribution. We argue such an approach scales gracefully with the size of the problem. On the theoretical side, this is supported by a guarantee that the number of planning oracle calls made scales inverse polynomially with the desired accuracy, but has no explicit dependence on the size of the state or the action space. Empirically, we demonstrate that such an approach to encouraging visitation of diverse states does indeed result in efficient task-agnostic exploration.

1.2 Previously published work

Chapter 2 contains work [ABH$^+$19] jointly conducted with Naman Agarwal, Brian Bullins, Elad Hazan and Sham Kakade, and the results therein appear in the proceedings of International Conference on Machine Learning (ICML) 2019.

Chapter 3 contains work [AHS19] jointly conducted with Naman Agarwal and Elad Hazan, and the results therein appear in the proceedings of Neural Information Processing Systems (NeurIPS) 2019.

Chapter 4 contains work [HKS20] jointly conducted with Elad Hazan and Sham Kakade, and the results therein appear in the proceedings of Algorithmic Learning Theory (ALT) 2020.

Chapter 5 contains work [AHMS21] jointly conducted with Naman Agarwal, Elad Hazan and Aniridha Majumdar, and the results therein appear in the proceedings of International Conference on Machine Learning (ICML) 2021.

Chapter 6 contains work [HS21] jointly conducted with Elad Hazan, and the results therein appear in the proceedings of International Conference on Machine Learning (ICML) 2021.

Chapter 7 contains work [HKSVS19] jointly conducted with Elad Hazan, Sham Kakade and Abby Van Soest, and the results therein appear in the proceedings of International Conference on Machine Learning (ICML) 2019.

Part I

The Nonstochastic Control Framework

Chapter 2

Online Control with Adversarial Disturbances

In this chapter, we study the control of linear dynamical systems with adversarial disturbances, as opposed to statistical noise. We present an efficient algorithm that achieves nearly-tight regret bounds in this setting. Our result generalizes upon previous work in two main aspects: the algorithm can accommodate adversarial noise in the dynamics, and can handle general convex costs.

2.1 Introduction

This chapter studies the robust control of linear dynamical systems. A linear dynamical system (LDS) is governed by the dynamics equation

$$x_{t+1} = Ax_t + Bu_t + w_t, \tag{2.1.1}$$

where x_t is the state, u_t is the control and w_t is a disturbance to the system. At every time step t, the controller suffers a cost $c_t(x_t, u_t)$ to enforce the control. In this chapter, we consider the setting of online control with *arbitrary* disturbances, under known transition dynamics specified by the matrices A and B. Formally, the setting involves, at every time step t, an adversary selecting a convex cost function $c_t(x, u)$ and a disturbance w_t, and the goal of the controller is to generate a sequence of controls u_t such that a sequence of convex costs $c_t(x_t, u_t)$ is minimized.

The above setting generalizes a fundamental problem in control theory, such as the Linear Quadratic Regulator, which has been studied over several decades. However, despite the significant

amount of literature on the problem, several challenges remained.

Challenge 1. Perhaps the most important challenge we address is in dealing with arbitrary disturbances w_t *in the dynamics*. This is a difficult problem, and so, the standard approaches often assume i.i.d. Gaussian noise. Worst-case approaches in the control literature, also known as H_∞-control and its variants, are considered overly pessimistic. Instead, we take an online (adaptive) approach to dealing with adversarial disturbances.

Challenge 2. Another limitation for efficient methods is the classical assumption that the costs $c_t(x, u)$ are quadratic, as is the case for the linear quadratic regulator. Part of the focus in the literature on the quadratic costs is due to special properties that allow for efficient computation of the best linear controller in hindsight via dynamic programming. One of our main goals is to introduce a more general technique that allows for efficient algorithms even when faced with arbitrary convex costs.

Our contributions. In this chapter, we tackle both of the challenges outlined above: coping with adversarial noise, and general loss functions in an online setting. For this we turn to the algorithmic methodology of regret minimization in online learning. To define the performance metric, we denote the cost for a control algorithm $\mathcal{A}$ as

$$J_T(\mathcal{A}) = \sum_{t=0}^{T} c_t(x_t, u_t).$$

The standard comparator in control is a linear controller, which generates a control signal as a linear function of the state, i.e., $u_t = -Kx_t$. Let $J(K)$ denote the cost of a linear controller from a certain class $K \in \mathcal{K}$. For an algorithm $\mathcal{A}$, we define the regret as the sub-optimality of its cost with respect to the best linear controller from a certain set, i.e.,

$$\texttt{Regret} = J_T(\mathcal{A}) - \min_{K \in \mathcal{K}} J_T(K).$$

Our main result is an efficient algorithm for control which achieves $O(\sqrt{T})$ regret in the setting described above. While a similar setting has been considered in the literature before [CHK$^+$18], our work generalizes previous work in the following ways:

1. Our algorithm achieves regret $O(\sqrt{T})$ even in the presence of bounded adversarial disturbances. Previous regret bounds needed to assume that the disturbances w_t are drawn from a distribution with zero mean and bounded variance.

2. Our regret bounds apply to any sequence of adversarially chosen convex loss functions. Previous efficient algorithms applied to convex quadratic costs only.

Our results above are obtained using several techniques from online convex optimization, notably online learning for loss functions with memory, and improper learning using convex relaxations.

2.1.1 Related work

Online Learning: This work advocates for worst-case regret as a robust performance metric in the presence of adversarial noise. A special case of our setting is that of regret minimization in stateless systems (where $A = 0$), which is a well-studied problem in machine learning. We refer the reader to various books and surveys on the topic [CBL06, SS+12, Haz16, ABC+19]. Of particular interest to our study is the setting of online learning with memory [AHM15].

Learning and Control in Linear Dynamical Systems: The modern setting for linear dynamical systems arose in the seminal work of Kalman [Kal60], who introduced the Kalman filter as a recursive least-squares solution for maximum likelihood estimation (MLE) of Gaussian perturbations to the system in latent-state systems. The framework and filtering algorithm have proven to be a mainstay in control theory and time-series analysis. We refer the reader to the classic survey [Lju98], and the extensive overview of recent literature in [HMR18]. Most of this literature, as well as much of classical control theory, deals with zero-mean random noise, typically normally distributed.

Recently, there has been a renewed interest in learning both fully-observable and latent-state linear dynamical systems. For fully-observable systems, sample complexity and regret bounds for control (under Gaussian noise) were obtained in [AYS11, DMM+18, AYLS19]. The technique of spectral filtering for learning and open-loop control of non-observable systems was introduced and studied in [HSZ17b, AHL+18, HLS+18, GLS+20, HSZ18]. Provable control in the Gaussian noise setting was also studied in [FGKM18].

Robust Control: A notable attempt to handle adversarial perturbations in the dynamics takes place in H_∞ control [Ste94a, ZDG+96b]. In this setting, the controller solves for the best linear controller assuming worst case noise to come, i.e.,

$$\min_{K_1} \max_{w_1} \min_{K_2} \ldots \min_{K_T} \max_{w_T} \sum_t c_t(x_t, u_t),$$

assuming similar linear dynamics. This approach is overly pessimistic, and leads to sub-optimal performance in various cases of interest. In comparison, we do not solve for the entire noise trajectory in advance, but adjust for it iteratively, and, in this manner, offer an instance-dependent guarantee. Another difference is computational: the above mathematical program may be hard to compute for general cost functions, as compared to our efficient gradient-based algorithm.

Non-stochastic MDPs: The setting we consider, namely control in systems with linear transition dynamics [Ber05] in the presence of adversarial disturbances, can be cast as that of planning in an adversarially changing MDP [ADT12, DH13]. The results obtained via this reduction are unsatisfactory because these regret bounds scale with the size of the state space, which is exponential in the dimension of the system. In addition, the regret in these scales as $\Omega(T^{\frac{2}{3}})$. In comparison, [YMS09, EDKM09] solve the online planning problem for MDPs with fixed dynamics and changing costs. The satisfying aspect of their result is that the regret bound does not explicitly depend on the size of the state space, and scales as $O(\sqrt{T})$. However, these assume that the dynamics are fixed and without (adversarial) noise.

LQR with Changing Costs: For the Linear Quadratic Regulator problem, [CHK$^+$18] consider changing quadratic costs with stochastic noise to achieve a $O(\sqrt{T})$ regret bound. This work is well aligned with our results, and the present chapter employs some notions developed therein (e.g., strong stability). However, the techniques used in [CHK$^+$18] (such as the SDP formulation for a linear controller) heavily rely on the quadratic nature of the cost functions and stochasticity of the disturbances. In particular, even for the offline problem, to the best of our knowledge, there does not exist an SDP formulation to determine the best linear controller for convex losses. In an earlier work, [AYS11] consider a more restricted setting with fixed, deterministic dynamics (hence, noiseless) and changing quadratic costs. [NK05] consider the adaptation of online learning to control in a manner that ensures stability. While the work does not provide any regret guarantee, the proposed notion of stability continues to be useful in the literature (e.g. [CHK$^+$18]) and this chapter.

Iterative Learning Control: We also note the similarity of our proposed algorithm to methods involving iterative learning control [ACM07, OH05] and feedback error learning [NS04]. These procedures, often operating over deterministic systems in the episodic setting, attempt to learn a control law via iteratively adjusting the controller as a function of the tracking error.

2.2 Problem setting

2.2.1 Interaction model

The linear dynamical system is a Markov decision process on continuous state and action spaces, with linear transition dynamics. In each round t, the learner outputs an action u_t upon observing the state x_t and incurs a cost of $c_t(x_t, u_t)$, where c_t is convex. The system then transitions to a new state x_{t+1} according to

$$x_{t+1} = Ax_t + Bu_t + w_t.$$

In the above definition, w_t is the disturbance the system suffers at each time step. In this chapter, we make no distributional assumptions on w_t, and the sequence w_t is not made known to the learner in advance.

For any algorithm $\mathcal{A}$, we attribute a cost of

$$J_T(\mathcal{A}) = \sum_{t=0}^{T} c_t(x_t, u_t),$$

where $x_{t+1} = Ax_t + Bu_t + w_t$ and $u_t = \mathcal{A}(x_1, \dots x_t)$. Overloading notation, we shall use $J(K)$ to denote the cost of a linear controller K which chooses the action as $u_t = -Kx_t$.

2.2.2 Assumptions

We make the following assumptions throughout the chapter. We remark that they are less restrictive than those considered by previous works, and hence allow for more general systems. In particular, we allow for adversarial (rather than i.i.d. stochastic) noise, and we also handle general convex cost functions. In addition, the non-stochastic nature of the disturbances permits, without loss of generality, the assumption that $x_0 = 0$.

Assumption 2.2.1. *The matrices that govern the dynamics are bounded, i.e., $\|A\| \le \kappa_A, \|B\| \le \kappa_B$. The perturbation introduced per time step is bounded, i.e., $\|w_t\| \le W$.*

Assumption 2.2.2. *The costs $c_t(x, u)$ are convex. Further, as long as it is guaranteed that $\|x\|, \|u\| \le D$, it holds that*

$$|c_t(x, u)| \le \beta D^2, \text{ and}$$

$$\|\nabla_x c_t(x, u)\|, \|\nabla_u c_t(x, u)\| \le GD.$$

Following the definitions in [CHK+18], we work on the following class of linear controllers.

Definition 2.2.3. *A linear policy K is (κ, γ)-strongly stable if there exist matrices L, Q satisfying $A - BK = QLQ^{-1}$, such that following two conditions are met:*

1. *The spectral norm of L is strictly smaller than one, i.e., $\|L\| \leq 1 - \gamma$.*
2. *The controller and the transforming matrices are bounded, i.e., $\|K\| \leq \kappa$ and $\|Q\|, \|Q^{-1}\| \leq \kappa$.*

2.2.3 Regret formulation

Let $\mathcal{K} = \{K : K \text{ is } (\kappa, \gamma)\text{-strongly stable}\}$. For an algorithm $\mathcal{A}$, the regret is the sub-optimality of its cost with respect to the best linear controller, i.e.,

$$\texttt{Regret} = J_T(\mathcal{A}) - \min_{K \in \mathcal{K}} J_T(K).$$

2.2.4 Proof techniques and overview

Choice of Policy Class: We begin by parameterizing the policy we execute at every step as a linear function of the disturbances in the past in Definition 2.3.1. Similar parameterization has been considered in the system level synthesis framework (see [WMD19]). This leads to a convex relaxation of the problem. Optimization on alternative parameterizations, including an SDP based framework [CHK$^+$18] or a direct parameterization [FGKM18], has been studied in the literature, but they seem unable to capture general convex functions as well as adversarial disturbance or lead to a non-convex loss. To avoid a linear dependence on time for the number of parameters in our policy, we additionally include a stable linear controller in our policy allowing us to effectively consider only $O(\gamma^{-1} \log(T))$ previous perturbations. Lemma 2.4.2 makes this notion of approximation precise.

Reduction to OCO with Memory: The choice of policy class with an appropriately chosen horizon H allows us to reduce the problem to compete with functions with truncated memory. This naturally falls under the class of online convex optimization with memory (see Section 2.3.5). Theorem 2.4.3 makes this reduction precise. Finally to bound the regret on truncated functions we use the Online Gradient Descent based approach specified in [AHM15], which requires a bound on Lipschitz constants which we provide in Section 2.4.3. This reduction is inspired by the ideas introduced in [EDKM09].

2.3 Preliminaries

In this section, we establish some important definitions that will prove to be useful throughout the chapter.

2.3.1 Notation

We reserve the letters x, y for states and u, v for control actions. We denote by $d = \max(\dim(x), \dim(u))$, i.e., a bound on the dimensionality of the problem. We reserve capital letters A, B, K, M for matrices associated with the system and the policy. Other capital letters are reserved for universal constants in the chapter. We use the shorthand $M_{i:j}$ to denote a subsequence $\{M_i, \ldots, M_j\}$.

2.3.2 A disturbance-action policy class

We establish the notion of a *disturbance-action controller* which chooses the action as a linear map of the past disturbances. Any disturbance-action controller ensures that the state of a system executing such a policy may be expressed as a linear function of the parameters of the policy. This property is convenient in that it permits efficient optimization over the parameters of such a policy. The situation may be contrasted with that of a linear controller. While the action recommended by a linear controller is also linear in past disturbances (a consequence of being linear in the current state), the state sequence produced on the execution of a linear policy is not a linear function of its parameters.

Definition 2.3.1 (Disturbance-Action Policy)**.** *A disturbance-action policy $\pi(M, K)$ is specified by parameters $M = (M^{[0]}, \ldots, M^{[H-1]})$ and a fixed matrix K, for horizon $H \geq 1$. At every time t, such a policy $\pi(M, K)$ chooses the recommended action u_t at a state x_t[1], defined as*

$$u_t = -Kx_t + \sum_{i=1}^{H} M^{[i-1]} w_{t-i}.$$

For notational convenience, here it may be considered that $w_i = 0$ for all $i < 0$.

We refer to the policy played at time t as $M_t = \{M_t^{[i]}\}$ where the subscript t refers to the time index and the superscript $[i]$ refers to the action of M_t on w_{t-i}. Note that such a policy can be executed because w_{t-1} is perfectly determined on the specification of x_t as $w_{t-1} = x_t - Ax_{t-1} - Bu_{t-1}$. It shall be established in later sections that such a policy class can approximate any linear policy with a strongly stable matrix in terms of the total cost suffered.

[1] x_t is completely determined given $w_0 \ldots w_{t-1}$. Hence, the use of x_t only serves to ease presentation.

2.3.3 Evolution of state

This section describes the evolution of the state of a linear dynamical system under a non-stationary policy $\pi = (\pi_0, \ldots, \pi_{T-1})$ composed of T policies, where each π_t is specified by $\pi_t(M_t = (M_t^{[0]}, \ldots, M_t^{[H-1]}), K)$. With some abuse of notation, we shall use $\pi((M_0, \ldots, M_{T-1}), K)$ to denote such a non-stationary policy.

The following definitions ease the burden of notation.

1. Define $\tilde{A}_K = A - BK$. $\tilde{A}_K$ shall be helpful in describing the evolution of state starting from a non-zero state in the absence of disturbances.

2. $x_t^K(M_{0:t-1})$ is the state attained by the system upon execution of a non-stationary policy $\pi(M_{0:t-1}, K)$. We similarly define $u_t^K(M_{0:t-1})$ to be the action executed at time t. If the same policy M is used across all time steps, we compress the notation to $x_t^K(M), u_t^K(M)$. Note that $x_t^K(0), u_t^K(0)$ refers to running the linear policy K.

3. $\Psi_{t,i}^{K,h}(M_{t-h:t})$ is a transfer matrix that describes the effect of w_{t-i} with respect to the past $h+1$ policies on the state x_{t+1}, formally defined below. When M is the same across all arguments we compress the notation to $\Psi_{t,i}^{K,h}(M)$.

Definition 2.3.2. *For any $t, h \le t, i \le H + h$, define the disturbance-state transfer matrix $\Psi_{t,i}^{K,h}$ to be a function with $h+1$ inputs defined as*

$$\Psi_{t,i}^{K,h}(M_{t-h:t}) = \tilde{A}_K^i \mathbf{1}_{i \le h} + \sum_{j=0}^{h} \tilde{A}_K^j B M_{t-j}^{[i-j-1]} \mathbf{1}_{i-j \in [1,H]}.$$

It will be worthwhile to note that $\Psi_{t,i}^{K,h}$ is linear in its arguments $M_{t-h:t}$. For the rest of the chapter we will set h to be H unless specified otherwise.

Lemma 2.3.3. *If u_t is chosen as what the non-stationary policy $\pi((M_0, \ldots, M_T), K)$ recommends, then the state sequence satisfies the following recurrence for any time t and $h \ge 0$:*

$$x_{t+1} = \tilde{A}_K^{h+1} x_{t-h} + \sum_{i=0}^{H+h} \Psi_{t,i}^{K,h}(M_{t-h:t}) w_{t-i}. \tag{2.3.1}$$

Proof. For any time t, we will prove the claim inductively on h. For $h = 0$, we have that

$$x_{t+1} = \tilde{A}_K x_t + \sum_{i=1}^{H} B M_t^{[i-1]} w_{t-i} + w_t = \tilde{A}_K x_t + \sum_{i=0}^{H} \Psi_{t,i}^{K,0}(M_t) w_{t-i}.$$

Further suppose the lemma holds for some $h \geq 0$, we prove it for $h+1$. We have that

$$\begin{aligned}
x_{t+1} &= \tilde{A}_K^{h+1} x_{t-h} + \sum_{i=0}^{H+h} \Psi_{t,i}^{K,h}(M_{t-h:t}) w_{t-i} \\
&= \tilde{A}_K^{h+1}(\tilde{A}_K x_{t-h-1} + \sum_{i=1}^{H} BM_{t-h-1}^{[i-1]} w_{t-h-i-1} + w_{t-h-1}) + \sum_{i=0}^{H+h} \Psi_{t,i}^{K,h}(M_{t-h:t}) w_{t-i} \\
&= \tilde{A}_K^{h+2} x_{t-h-1} + \sum_{i=0}^{H+h} \left(\Psi_{t,i}^{K,h}(M_{t-h:t}) + \tilde{A}_K^{i} \mathbf{1}_{i=h+1} + \tilde{A}_K^{h+1} BM_{t-h-1}^{[i-h-2]} \mathbf{1}_{i-h-1 \in [1,H]} \right) w_{t-i} \\
&= \tilde{A}_K^{h+2} x_{t-h-1} + \sum_{i=0}^{H+h+1} \Psi_{t,i}^{K,h+1}(M_{t-h-1:t}) w_{t-i}.
\end{aligned}$$

The proof now follows by induction. □

2.3.4 Idealized setting

Note that the counter-factual nature of regret in the control setting implies in the loss at a time step t, depends on all the choices made in the past. To efficiently deal with this we propose that our optimization problem only consider the effect of the past H steps while planning, forgetting about the state, the system was at time $t-H$. We will show later that the above scheme tracks the true cost suffered upto a small additional loss. To formally define this idea, we need the following notion of an *ideal* state.

Definition 2.3.4 (Ideal State & Action)**.** *Define an ideal state $y_{t+1}^K(M_{t-H:t})$ which is the state the system would have reached if it played the non-stationary policy $M_{t-H:t}$ at all time steps from $t-H$ to t, assuming the state at $t-H$ is 0. Similarly, define $v_{t+1}^K(M_{t-H:t+1})$ to be an idealized action that would have been executed at time $t+1$ if the state observed at time $t+1$ is $y_{t+1}^K(M_{t-H:t})$. Formally,*

$$\begin{aligned}
y_{t+1}^K(M_{t-H:t}) &= \sum_{i=0}^{2H} \Psi_{t,i}^{K,H}(M_{t-H:t}) w_{t-i}, \\
v_{t+1}^K(M_{t-H:t+1}) &= -K y_{t+1}^K(M_{t-H:t}) + \sum_{i=1}^{H} M_{t+1}^{[i-1]} w_{t+1-i}.
\end{aligned}$$

When M is the same across all arguments we compress the notation to $y_{t+1}^K(M), v_{t+1}^K(M)$.

We can now consider the loss of the *ideal* state and the *ideal* action.

Definition 2.3.5 (Ideal Cost)**.** *Define the idealized cost function f_t to be the cost associated with*

the idealized state and idealized action, i.e.,

$$f_t(M_{t-H-1:t}) = c_t(y_t^K(M_{t-H-1:t-1}), v_t^K(M_{t-H-1:t})).$$

When M is the same across all arguments we compress the notation to $f_t(M)$.

The linearity of y_t^K in past controllers and the linearity of v_t^K in its immediate state implies that f_t is a convex function of a linear transformation of $M_{t-H-1:t}$ and hence convex in $M_{t-H-1:t}$. This renders it amenable to algorithms for online convex optimization.

In Theorem 2.4.3 we show that for a given sequence of policies $\{M_i\}$, the idealized cost f_t and the real cost c_t are close by and this reduction allows us to only consider the truncated f_t while planning, hence allowing for efficiency. The precise notion of minimizing regret on such truncated f_t was considered in online learning literature before as online convex optimization (OCO) with memory [AHM15]. We present an overview of this framework next.

2.3.5 OCO with memory

We now present an overview of the online convex optimization (OCO) with memory framework, as established by [AHM15]. In particular, we consider the setting where, for every t, an online player chooses some point $x_t \in \mathcal{K} \subset \mathbb{R}^d$, a loss function $f_t : \mathcal{K}^{H+1} \mapsto \mathbb{R}$ is revealed, and the learner suffers a loss of $f_t(x_{t-H:t})$. We assume a certain coordinate-wise Lipschitz regularity on f_t of the form such that, for any $j \in \{0, \ldots, H\}$, for any $x_{0:H}, \tilde{x}_j \in \mathcal{K}$,

$$|f_t(x_{0:j-1}, x_j, x_{j+1:H}) - f_t(x_{0:j-1}, \tilde{x}_j, x_{j+1:H})| \leq L\|x_j - \tilde{x}_j\|. \tag{2.3.2}$$

In addition, we define $\tilde{f}_t(x) = f_t(x, \ldots, x)$, and we let

$$G_f = \sup_{t \in \{0,\ldots,T\}, x \in \mathcal{K}} \|\nabla \tilde{f}_t(x)\|, \quad D = \sup_{x,y \in \mathcal{K}} \|x - y\|. \tag{2.3.3}$$

The resulting goal is to minimize the *policy regret* [ADT12], which is defined as

$$\texttt{PolicyRegret} = \sum_{t=H}^{T} f_t(x_{t-H:t}) - \min_{x \in \mathcal{K}} \sum_{t=0}^{T} \tilde{f}_t(x).$$

As shown by [AHM15], by running a memory-based OGD, we may bound the policy regret by the following theorem.

Algorithm 1 Online Control Algorithm

Input: Step size η, Control Matrix K, Parameters $\kappa_B, \kappa, \gamma, T$.
Define $H = \gamma^{-1}\log(T\kappa^2)$
Define $\mathcal{M} = \{M = \{M^{[0]} \dots M^{[H-1]}\} : \|M^{[i-1]}\| \leq \kappa^3\kappa_B(1-\gamma)^i\}$.
Initialize $M_0 \in \mathcal{M}$ arbitrarily.
for $t = 0, \dots, T-1$ **do**
    Choose the action:
$$u_t = c_t - Kx_t + \sum_{i=1}^{H} M_t^{[i-1]} w_{t-i}.$$
    Observe the new state x_{t+1} and record $w_t = x_{t+1} - Ax_t - Bu_t$.
    Set $M_{t+1} = \Pi_{\mathcal{M}}(M_t - \eta\nabla f_t(M_t))$
end for

Theorem 2.3.6. *Let $\{f_t\}_{t=H}^{T}$ be Lipschitz continuous loss functions with memory such that $\tilde{f}_t$ are convex, and let L, D, and G_f be as defined in equations 2.3.2 and 2.3.3. Then, there exists an algorithm which generates a sequence $\{x_t\}_{t=0}^{T}$ such that*

$$\sum_{t=H}^{T} f_t(x_{t-H:t}) - \min_{x\in\mathcal{K}} \sum_{t=H}^{T} \tilde{f}_t(x) \leq 3D\sqrt{G_f(G_f + LH^2)T}.$$

2.4 Algorithm & main result

Algorithm 1 describes our proposed algorithm for controlling linear dynamical systems with adversarial disturbances which at all times maintains a disturbance-action controller. The algorithm implements the memory based OGD on the loss $f_t(\cdot)$ as described in the previous section. The algorithm requires the specification of a (κ, γ)-strongly stable matrix K once before the online game. Such a matrix can be obtained offline using an SDP relaxation as described in [CHK$^+$18]. The following theorem states the regret bound Algorithm 1 guarantees.

Theorem 2.4.1 (Main Theorem)**.** *Suppose Algorithm 1 is executed with $\eta = \Theta\left(GW\sqrt{T}\right)^{-1}$, on an LDS satisfying Assumption 2.2.1 with control costs satisfying Assumption 2.2.2. Then, it holds true that*

$$J_T(\mathcal{A}) - \min_{K\in\mathcal{K}} J_T(K) \leq O\left(GW^2\sqrt{T}\log(T)\right),$$

Furthermore, the algorithm maintains at most $O(1)$ parameters and can be implemented in time $O(1)$ per time step. Here $O(\cdot)$, $\Theta(\cdot)$ contain polynomial factors in $\gamma^{-1}, \kappa_B, \kappa, d$.

Proof of Theorem 2.4.1. Note that by the definition of the algorithm we have that all $M_t \in \mathcal{M}$, where

$$\mathcal{M} = \{M = \{M^{[0]} \dots M^{[H-1]}\} : \|M^{[i-1]}\| \leq \kappa^3\kappa_B(1-\gamma)^i\}.$$

Let D be defined as

$$D \triangleq \frac{W\kappa^3(1+H\kappa_B\tau)}{\gamma(1-\kappa^2(1-\gamma)^{H+1})} + \frac{\kappa_B\kappa^3 W}{\gamma}.$$

Let K^* be the optimal linear policy in hindsight. By definition K^* is a (κ,γ)-strongly stable matrix. Using Lemma 2.4.2 and Theorem 2.4.3, we have that

$$\begin{aligned}
&\min_{M_*\in\mathcal{M}}\left(\sum_{t=0}^{T} f_t(M_*)\right) - \sum_{t=0}^{T} c_t(x_t^{K^*}(0), u_t^{K^*}(0)) \\
&\quad\le \min_{M_*\in\mathcal{M}}\left(\sum_{t=0}^{T} c_t(x_t^{K}(M_*), u_t^{K}(M_*))\right) - \sum_{t=0}^{T} c_t(x_t^{K^*}(0), u_t^{K^*}(0)) + 2TGD^2\kappa^3(1-\gamma)^{H+1} \\
&\quad\le 2TGD(1-\gamma)^{H+1}\left(\frac{WH\kappa_B^2\kappa^5}{\gamma} + D\kappa^3\right). \qquad (2.4.1)
\end{aligned}$$

Let $M_1 \dots M_T$ be the sequence of policies played by the algorithm. Note that by definition of the constraint set S, we have that

$$\forall t \in [T], \forall i \in [H] \quad \|M_t^{[i]}\| \le \kappa_B\kappa^3(1-\gamma)^i.$$

Using Theorem 2.4.3 we have that

$$\sum_{t=0}^{T} c_t(x_t^K(M_{0:t-1}), u_t^K(M_{0:t-1})) - \sum_{t=0}^{T} f_t(M_{t-H-1:t}) \le 2TGD^2\kappa^3(1-\gamma)^{H+1}. \qquad (2.4.2)$$

Finally using Theorem 2.3.6 and using Lemmas 2.4.6, 2.4.7 to bound the constants G_f and L associated with the function f_t and by noting that

$$\max_{M_1,M_2\in\mathcal{M}} \|M_1 - M_2\| \le \frac{\kappa_B\kappa^3\sqrt{d}}{\gamma},$$

we have that

$$\sum_{t=0}^{T} f_t(M_{t-H-1:t}) - \min_{M_*\in\mathcal{M}} \sum_{t=0}^{T} f_t(M_*) \le 8GWDd^{3/2}\kappa_B^2\kappa^6 H^{2.5}\gamma^{-1}\sqrt{T}. \qquad (2.4.3)$$

Summing up equations 2.4.1, 2.4.2 and 2.4.3, and using the condition that $H = \frac{1}{\gamma}\log(T\kappa^2)$, we get the result. □

2.4.1 Sufficiency of disturbance-action policies

The class of policies described in Definition 2.3.1 is powerful enough to capture any fixed linear policy. Lemma 2.4.2 establishes this equivalence in terms of the state and action sequence that each policy produces.

Lemma 2.4.2 (Sufficiency)**.** *For any two (κ, γ)-strongly stable matrices K^*, K, there exists a policy $\pi(M_*, K)$, with $M_* = (M_*^{[0]}, \ldots, M_*^{[H-1]})$ defined as*

$$M_*^{[i]} = (K - K^*)(A - BK^*)^i$$

such that

$$\sum_{t=0}^{T} \left(c_t(x_t^K(M_*), u_t^K(M_*)) - c_t(x_t^{K^*}(0), u_t^{K^*}(0)) \right) \leq T \cdot \frac{2GDWH\kappa_B^2\kappa^5(1-\gamma)^H}{\gamma}.$$

Proof of Lemma 2.4.2. By definition we have that

$$x_{t+1}^{K^*}(0) = \sum_{i=0}^{t} \tilde{A}_{K^*}^i w_{t-i}.$$

Consider the following calculation for M_* with $M_*^{[i]} \triangleq (K-K^*)(A-BK^*)^i$ and for any $i \in \{0 \ldots H\}$. We have for any $h \geq H$ that

$$\begin{aligned}
\Psi_{t,i}^{K,h}(M_*) &= \tilde{A}_K^i + \sum_{j=1}^{i} \tilde{A}_K^{i-j} B M_*^{[j-1]} \\
&= \tilde{A}_K^i + \sum_{j=1}^{i} \tilde{A}_K^{i-j} B(K-K^*)\tilde{A}_{K^*}^{j-1} \\
&= \tilde{A}_K^i + \sum_{j=1}^{i} \tilde{A}_K^{i-j} (\tilde{A}_{K^*} - \tilde{A}_K)\tilde{A}_{K^*}^{j-1} \\
&= \tilde{A}_K^i + \sum_{j=1}^{i} \left(\tilde{A}_K^{i-j}\tilde{A}_{K^*}^{j} - \tilde{A}_K^{i-j+1}\tilde{A}_{K^*}^{j-1} \right) \\
&= \tilde{A}_{K^*}^i
\end{aligned}$$

The final equality follows as the sum telescopes. Therefore, we have that

$$x_{t+1}^K(M_*) = \sum_{i=0}^{H} \tilde{A}_{K^*}^i w_{t-i} + \sum_{i=H+1}^{t} \Psi_{t,i}^{K,t}(M_*) w_{t-i}.$$

From the above we get that

$$\|x_t^{K^*}(0) - x_t^K(M_*)\| \le W\left(\sum_{i=H+1}^{t} \|\Psi_{t,i}^{K,t}(M_*)\| + \sum_{i=H+1}^{t} \|\tilde{A}_{K^*}^i\|\right) \le \frac{2WH\kappa_B^2\kappa^5(1-\gamma)^H}{\gamma},$$

where the last inequality follows from using Lemma 2.4.4 and using the fact that $\|M_*^{[i]}\| \le \kappa_B\kappa^3(1-\gamma)^i$. Further comparing the actions taken by the two policies, we may see that

$$\begin{aligned}
\|u_t^{K^*} - u_t^K(M_*)\| &= \left\| -K^* x_t^{K^*} + K x_t^K(M_*) - \sum_{i=1}^{H} (K^* - K)\tilde{A}_{K^*}^{i-1} w_{t-i} \right\| \\
&\le \left\| \sum_{i=H+1}^{t} K\left(\tilde{A}_{K^*}^i + \Psi_{t,i}^K(M_*)\right) w_{t-i} \right\| \\
&\le \frac{2WH\kappa_B^2\kappa^5(1-\gamma)^H}{\gamma}.
\end{aligned}$$

Using the above, Assumption 2.2.2 and Lemma 2.4.5, we get that

$$\sum_{t=0}^{T} \left(c_t(x_t^K(M_*), u_t^K(M_*)) - c_t(x_t^{K^*}(0), u_t^{K^*}(0))\right) \le T \cdot \frac{2GDWH\kappa_B^2\kappa^5(1-\gamma)^H}{\gamma}. \qquad \square$$

2.4.2 Approximation theorems

The following theorem relates the cost of $f_t(M_{t-H-1:t})$ with the actual cost $c_t(x_t^K(M_{0:t-1}), u_t^K(M_{0:t}))$.

Theorem 2.4.3. *For any (κ,γ)-strongly stable K, any $\tau > 0$, and any sequence of policies $M_1 \dots M_T$ satisfying $\|M_t^{[i]}\| \le \tau(1-\gamma)^i$, if the perturbations are bounded by W, we have that*

$$\sum_{t=1}^{T} f_t(M_{t-H-1:t}) - \sum_{t=1}^{T} c_t(x_t^K(M_{0:t-1}), u_t^K(M_{0:t})) \le 2TGD^2\kappa^3(1-\gamma)^{H+1},$$

where

$$D \triangleq \frac{W\kappa^3(1+H\kappa_B\tau)}{\gamma(1-\kappa^2(1-\gamma)^{H+1})} + \frac{\tau W}{\gamma}.$$

Before proving the above theorem, we will need a few lemmas which will be useful.

Lemma 2.4.4. *Let K be a (κ,γ)-strongly stable matrix, $\tau > 0$, and M_t be a sequence such that for all i,t, we have $\|M_t^{[i]}\| \le \tau(1-\gamma)^i$. Then we have that for all i,t,h,*

$$\|\Psi_{t,i}^{K,h}\| \le \kappa^2(1-\gamma)^i \cdot \mathbf{1}_{i\le H} + H\kappa_B\kappa^2\tau(1-\gamma)^{i-1}.$$

We now derive a bound on the norm of each of the states.

Lemma 2.4.5. *Suppose the system satisfies Assumption 2.2.1 and let M_t be a sequence such that for all i, t, we have that $\|M_t^{[i]}\| \leq \tau(1-\gamma)^i$ for a number τ. Define*

$$D \triangleq \frac{W\kappa^3(1+H\kappa_B\tau)}{\gamma(1-\kappa^2(1-\gamma)^{H+1})} + \frac{aW}{\gamma}.$$

Further suppose K^* is a (κ,γ)-strongly stable matrix. We have that for all t,

$$\|x_t^K(M_{0:t-1})\|, \|y_t^K(M_{t-H-1:t-1})\|, \|x_t^{K^*}(0)\| \leq D$$

$$\|u_t^K(M_{0:t-1})\|, \|v_t^K(M_{t-H-1:t})\| \leq D$$

$$\|x_t^K(M_{0:t-1}) - y_t^K(M_{t-H-1:t-1})\| \leq \kappa^2(1-\gamma)^{H+1}D$$

$$\|u_t^K(M_{0:t}) - v_t^K(M_{t-H-1:t})\| \leq \kappa^3(1-\gamma)^{H+1}D.$$

Finally, we prove Theorem 2.4.3.

Proof of Theorem 2.4.3. Using the above lemmas we can now bound the approximation error between f_t and c_t using Assumption 2.2.2

$$\begin{aligned}
&|c_t(x_t^K(M_{0:t-1}),u_t^K(M_{0:t})) - f_t(M_{t-H-1:t})| \\
&\qquad = |c_t(x_t^K(M_{0:t}), u_t^K(M_{0:t-1})) - c_t(y_t^K(M_{t-H-1:t-1}), v_t^K(M_{t-H-1:t}))| \\
&\qquad \leq GD\|x_t^K(M_{0:t-1}) - y_t^K(M_{t-H-1:t-1})\| + GD\|u_t^K(M_{0:t}) - v_t^K(M_{t-H-1:t}))\| \\
&\qquad \leq 2GD^2\kappa^3(1-\gamma)^{H+1}.
\end{aligned}$$

This finishes the proof of Theorem 2.4.3. □

2.4.3 Bounding the properties of OCO with memory

Thus far, Lemma 2.4.2 establishes that it is sufficient to compare against the class of disturbance-action policies. For such policies, Theorem 2.4.3 reduces the counter-factual regret minimization problem to online learning on loss functions with memory. It remains to quantify the constants with which Theorem 2.3.6 may be invoked to obtain a regret bound. Note that Line 3 of Algorithm 1 places on upper bound on the diameter D.

Bounding the Lipschitz Constant

We begin by establishing an upper bound on the lipschitz constant L as defined in Equation 2.3.2.

Lemma 2.4.6. *Consider two policy sequences* $\{M_{t-H-1} \dots M_{t-k} \dots M_t\}$ *and* $\{M_{t-H-1} \dots \tilde{M}_{t-k} \dots M_t\}$ *which differ in exactly one policy played at a time step* $t-k$ *for* $k \in \{0, \dots, H\}$*. Then we have that*

$$|f_t(M_{t-H-1} \dots M_{t-k} \dots M_t) - f_t(M_{t-H-1} \dots \tilde{M}_{t-k} \dots M_t)| \leq 2GDW\kappa_B\kappa^3(1-\gamma)^k \sum_{i=0}^{H} \left(\|M_{t-k}^{[i]} - \tilde{M}_{t-k}^{[i]}\| \right).$$

Bounding the Gradient

A bound on the norm of the gradient follows similarly.

Lemma 2.4.7. *For all* M *such that* $\|M^{[j]}\| \leq \tau(1-\gamma)^j$ *for all* $j \in [0, H-1]$*, we have that*

$$\|\nabla_M f_t(M)\|_F \leq GDWHd\left(\frac{2\kappa_B\kappa^3}{\gamma} + H\right).$$

Note that since M is a matrix, the ℓ_2 norm of the gradient $\nabla_M f_t$ corresponds to to the Frobenius norm of the $\nabla_M f_t$ matrix.

2.5 Remaining proofs

2.5.1 Proof of Theorem 2.3.6

Proof. By the standard OGD analysis, we know that

$$\sum_{t=H}^{T} \tilde{f}_t(x_t) - \min_{x \in \mathcal{K}} \sum_{t=H}^{T} \tilde{f}_t(x) \leq \frac{D^2}{\eta} + TG^2\eta.$$

In addition, we know by equation 2.3.2 that, for any $t \geq H$,

$$\begin{aligned} |f_t(x_{t-H}, \dots, x_t) - f_t(x_t, \dots, x_t)| &\leq L\sum_{j=1}^{H} \|x_t - x_{t-j}\| \leq L\sum_{j=1}^{H}\sum_{l=1}^{j} \|x_{t-l+1} - x_{t-l}\| \\ &\leq L\sum_{j=1}^{H}\sum_{l=1}^{j} \eta\|\nabla \tilde{f}_{t-l}(x_{t-l})\| \leq LH^2\eta G, \end{aligned}$$

and so we have that

$$\left| \sum_{t=H}^{T} f_t(x_{t-H}, \dots, x_t) - \sum_{t=H}^{T} f_t(x_t, \dots, x_t) \right| \leq TLH^2\eta G.$$

Algorithm 2 OGD with Memory (OGD-M).

Input: Step size η, functions $\{f_t\}_{t=m}^{T}$
Initialize $x_0, \ldots, x_{H-1} \in \mathcal{K}$ arbitrarily.
for $t = H, \ldots, T$ **do**
    Play x_t, suffer loss $f_t(x_{t-H}, \ldots, x_t)$
    Set $x_{t+1} = \Pi_{\mathcal{K}} \left(x_t - \eta \nabla \tilde{f}_t(x) \right)$
end for

It follows that

$$\sum_{t=H}^{T} f_t(x_{t-H}, \ldots, x_t) - \min_{x \in \mathcal{K}} \sum_{t=H}^{T} f_t(x, \ldots, x) \leq \frac{D^2}{\eta} + TG_f^2\eta + LH^2\eta G_f T.$$

The theorem follows after setting $\eta = \frac{D}{\sqrt{G_f(G_f + LH^2)T}}$. □

2.5.2 Proof of Lemma 2.4.4

Proof of Lemma 2.4.4. The proof follows by noticing that

$$\begin{aligned}
\|\Psi_{t,i}^{K}\| &\leq \|\tilde{A}_K^i\| \mathbf{1}_{i \leq H} + \sum_{j=1}^{H} \|\tilde{A}_K^j\| \|B\| \|M_{t-j}^{[i-j]}\| \mathbf{1}_{i-j \in [1,H]} \\
&\leq \kappa^2 (1-\gamma)^i \cdot \mathbf{1}_{i \leq H} + \sum_{j=1}^{H} \kappa_B \kappa^2 \tau (1-\gamma)^{i-1} \\
&\leq \kappa^2 (1-\gamma)^i \cdot \mathbf{1}_{i \leq H} + H\kappa_B \kappa^2 \tau (1-\gamma)^{i-1},
\end{aligned}$$

where the second and the third inequalities follow by using the fact that K is a (κ, γ)-strongly stable matrix and the conditions on the spectral norm of M. □

2.5.3 Proof of Lemma 2.4.5

Proof of Lemma 2.4.5. Using the definition of x_t we have that

$$\begin{aligned}
\|x_t^K(M_{0:t-1})\| &\leq \kappa^2 (1-\gamma)^{H+1} \|x_{t-H-1}^K(M_{0:t-H-2})\| + W \cdot \left(\sum_{i=0}^{2H} \|\Psi_{t-1,i}^{K,H}(M_{t-H,t-1})\| \right) \\
&\leq \kappa^2 (1-\gamma)^{H+1} \|x_{t-H-1}(M_{0:t-H-2})\| + W \cdot \left(\frac{\kappa^2 + H\kappa_B\kappa^2\tau}{\gamma} \right).
\end{aligned}$$

The above recurrence can be seen to easily satisfy the following upper bound:

$$\|x_t^K(M_{0:t-1})\| \leq \frac{W(\kappa^2 + H\kappa_B\kappa^2 a)}{\gamma(1 - \kappa^2(1-\gamma)^{H+1})} \leq D. \tag{2.5.1}$$

A similar bound can be established for

$$\|y_t^K(M_{t-H-1:t-1})\| \leq W \cdot \left(\frac{\kappa^2 + H\kappa_B\kappa^2 a}{\gamma}\right) \leq D. \tag{2.5.2}$$

It is also simple to see via the definitions that

$$\|x_t^K - y_t^K(M_{t-H-1} \dots M_{t-1})\| \leq \|\tilde{A}_K^i\| \|x_{t-H}\| \leq \kappa^2(1-\gamma)^{H+1} D \tag{2.5.3}$$

We can now finally bound

$$\|x_t^{K^*}(0)\| \leq \frac{W\kappa^2}{\gamma} \leq D.$$

For the actions we can use the definitions to bound the actions as follows using equations 2.5.1 and 2.5.2

$$\|u_t^K(M_{0:t})\| \leq \|Kx_t^K(M_{0:t-1})\| + \sum_{i=1}^{H} \|M_t^{[i-1]} w_{t-i}\| \leq \kappa \|x_t^K(M_{0:t-1})\| + \frac{\tau W}{\gamma} \leq D,$$

$$\|v_t^K(M_{t-H} \dots M_t)\| \leq \|Ky_t^K(M_{t-H-1} \dots M_{t-1})\| + \sum_{i=1}^{H} \|M_t^{[i]} w_{t-i}\| \leq D.$$

We also have that using equation 2.5.3

$$\|u_t^K(M_{t-H-1:t}) - v_t^K(M_{t-H-1:t})\| = K(x_t^K(M_{0:t-1}) - y_t^K(M_{t-H-1:t-1})) \leq \kappa^3(1-\gamma)^{H+1} D.$$

□

2.5.4 Proof of Lemma 2.4.6

Proof of Lemma 2.4.6. For the rest of the proof, we will denote $y_{t+1}^K(\{M_{t-H-1} \dots M_{t-k} \dots M_{t-1}\})$ as y_{t+1}^K and $y_{t+1}^K(\{M_{t-H-1} \dots \tilde{M}_{t-k} \dots M_{t-1}\})$ as $\tilde{y}_{t+1}^K$. Similarly define v_t^K and $\tilde{v}_t^K$. It follows immediately from the definitions that

$$\begin{aligned}
\|y_t^K - \tilde{y}_t^K\| &= \|\tilde{A}_K^k B \sum_{i=0}^{2H} \left(M_{t-k}^{[i-k]} - \tilde{M}_{t-k}^{[i-k]}\right) w_{t-i} \mathbf{1}_{i-k \in [1,H]}\| \\
&\leq \kappa_B \kappa^2 (1-\gamma)^k W \sum_{i=0}^{H} \left(\|M_{t-k}^{[i]} - \tilde{M}_{t-k}^{[i]}\|\right).
\end{aligned}$$

Furthermore, we have that

$$\begin{aligned}\|v_t^K - \tilde{v}_t^K\| &= \| - K(y_t - \tilde{y}_t) + \mathbf{1}_{k=0}\sum_{i=0}^{H}(M_t^{[i]} - \tilde{M}_t^{[i]})w_{t-i}\| \\ &\leq 2\kappa_B\kappa^3(1-\gamma)^k W\sum_{i=0}^{H}\left(\|M_{t-k}^{[i]} - \tilde{M}_{t-k}^{[i]}\|\right). \qquad \square\end{aligned}$$

Therefore, using Assumption 2.2.2 and Lemma 2.4.5, we immediately get that

$$f_t(M_{t-H}\dots M_{t-k}\dots M_t) - f_t(M_{t-H}\dots \tilde{M}_{t-k}\dots M_t) \leq 2GDW\kappa_B\kappa^3(1-\gamma)^k\sum_{i=0}^{H}\left(\|M_{t-k}^{[i-1]} - \tilde{M}_{t-k}^{[i-1]}\|\right).$$

2.5.5 Proof of Lemma 2.4.7

Proof of Lemma 2.4.7. To derive a crude bound on the quantity in question, it will be sufficient to derive an absolute value bound on $\nabla_{M_{p,q}^{[r]}} f_t(M)$ for all r, p, q. To this end, we consider the following calculation. Using Lemma 2.4.5, we get that $y_t^K(M), v_t^K(M) \leq D$. Therefore, using Assumption 2.2.2, we have that

$$|\nabla_{M_{p,q}^{[r]}} f_t(M)| \leq GD\left(\left\|\frac{\partial y_t^K(M)}{\partial M_{p,q}^{[r]}} + \frac{\partial v_t^K(M)}{\partial M_{p,q}^{[r]}}\right\|\right).$$

We now bound the quantities on the right-hand side:

$$\left\|\frac{\partial y_t^K(M)}{\partial M_{p,q}^{[r]}}\right\| = \left\|\sum_{i=0}^{2H}\sum_{j=0}^{H}\left[\frac{\partial \tilde{A}_K^j BM^{[i-j-1]}}{\partial M_{p,q}^{[r]}}\right]w_{t-i}\mathbf{1}_{i-j\in[1,H]}\right\| \leq \sum_{i=r+1}^{r+H+1}\left\|\left[\frac{\partial \tilde{A}_K^{i-r-1}BM^{[r]}}{\partial M_{p,q}^{[r]}}\right]w_{t-i}\right\| \leq \frac{W\kappa_B\kappa^2}{\gamma}.$$

Similarly,

$$\left\|\frac{\partial v_t^K(M)}{\partial M_{p,q}^{[r]}}\right\| \leq \kappa\left\|\frac{\delta y_t^K(M)}{\delta M_{p,q}^{[r]}}\right\| + \left\|\sum_{i=1}^{H}\frac{\partial M^{[i-1]}}{\partial M_{p,q}^{[r]}}w_{t-i}\right\| \leq W\left(\frac{\kappa_B\kappa^3}{\gamma} + H\right).$$

Combining the above inequalities gives the bound in the lemma. $\square$

Chapter 3

Logarithmic Regret for Online Control

This chapter studies optimal regret bounds for control in linear dynamical systems under adversarially changing strongly convex cost functions, given the knowledge of transition dynamics. This includes several well studied and fundamental frameworks such as the Kalman filter and the linear quadratic regulator. State of the art methods achieve regret which scales as $O(\sqrt{T})$, where T is the time horizon.

We show that the optimal regret in this setting can be significantly smaller, scaling as $O(\text{poly}(\log T))$. This regret bound is achieved by two different efficient iterative methods, online gradient descent and online natural gradient.

3.1 Introduction

Algorithms for regret minimization typically attain one of two performance guarantees. For general convex losses, regret scales as square root of the number of iterations, and this is tight. However, if the loss function exhibit more curvature, such as quadratic loss functions, there exist algorithms that attain poly-logarithmic regret. This distinction is also known as "fast rates" in statistical estimation.

Despite their ubiquitous use in online learning and statistical estimation, logarithmic regret algorithms are almost non-existent in control of dynamical systems. This can be attributed to fundamental challenges in computing the optimal controller in the presence of noise.

Time-varying cost functions in dynamical systems can be used to model unpredictable dynamic

resource constraints, and the tracking of a desired sequence of exogenous states. At a pinch, if we have changing (even, strongly) convex loss functions, the optimal controller for a linear dynamical system is not immediately computable via a convex program. For the special case of quadratic loss, some previous works [CHK+18] remedy the situation by taking a semi-definite relaxation, and thereby obtain a controller which has provable guarantees on regret and computational requirements. However, this semi-definite relaxation reduces the problem to regret minimization over linear costs, and removes the curvature which is necessary to obtain logarithmic regret.

In this chapter we give the first efficient poly-logarithmic regret algorithms for controlling a linear dynamical system with noise in the dynamics (i.e. the standard model). Our results apply to general convex loss functions that are strongly convex, and not only to quadratics.

Reference	Noise	Regret	loss functions
[AYBK14]	**none**	$O(\log^2 T)$	quadratic (fixed hessian)
[ABH+19]	adversarial	$O(\sqrt{T})$	convex
[CHK+18]	stochastic	$O(\sqrt{T})$	quadratic
here	stochastic	$O(\log^7 T)$	strongly convex

3.1.1 Statement of results

The setting we consider is a linear dynamical system, a continuous state Markov decision process with linear transitions, described by the following equation:

$$x_{t+1} = Ax_t + Bu_t + w_t. \tag{3.1.1}$$

Here x_t is the state of the system, u_t is the action (or control) taken by the controller, and w_t is the noise. In each round t, the learner outputs an action u_t upon observing the state x_t and incurs a cost of $c_t(x_t, u_t)$, where c_t is convex. The objective here is to choose a sequence of adaptive controls u_t so that a minimum total cost may be incurred.

The approach taken by [CHK+18] and other previous works is to use a semi-definite relaxation for the controller. However, this removes the properties associated with the curvature of the loss functions, by reducing the problem to an instance of online linear optimization. It is known that without curvature, $O(\sqrt{T})$ regret bounds are tight (see [Haz16]).

Therefore we take a different approach, initiated by [ABH+19]. We consider controllers that depend on the previous noise terms, and take the form $u_t = \sum_{i=1}^{H} M_i w_{t-i}$. While this resulting convex relaxation does not remove the curvature of the loss functions altogether, it results in an

overparametrized representation of the controller, and it is not a priori clear that the loss functions are strongly convex with respect to the parameterization. We demonstrate the appropriate conditions on the linear dynamical system under which the strong convexity is retained.

Henceforth we present two methods that attain poly-logarithmic regret. They differ in terms of the regret bounds they afford and the computational cost of their execution. The online gradient descent update (OGD) requires only gradient computation and update, whereas the online natural gradient (ONG) update, in addition, requires the computation of the preconditioner, which is the expected Gram matrix of the Jacobian, denoted J, and its inverse. However, the natural gradient update admits an instance-dependent upper bound on the regret, which while being at least as good as the regret bound on OGD, offers better guarantees on benign instances (See Corollary 3.4.5, for example).

Algorithm	Update rule (simplified)	Applicability
OGD	$M_{t+1} \leftarrow M_t - \eta_t \nabla f_t(M_t)$	$\exists K$, diag L s.t. $A - BK = QLQ^{-1}$
ONG	$M_{t+1} \leftarrow M_t - \eta_t (\mathbb{E}[J^\top J])^{-1} \nabla f_t(M_t)$	$\|L\| \leq 1 - \delta$, $\|Q\|, \|Q\|^{-1} \leq \kappa$

3.1.2 Related work

For a survey of linear dynamical systems (LDS), as well as learning, prediction and control problems, see [Ste94b]. Recently, there has been a renewed interest in learning dynamical systems in the machine learning literature. For fully-observable systems, sample complexity and regret bounds for control (under Gaussian noise) were obtained in [AYS11, DMM$^+$18, AYLS19]. The technique of spectral filtering for learning and open-loop control of partially observable systems was introduced and studied in [HSZ17b, AHL$^+$18, HLS$^+$18]. Provable control in the Gaussian noise setting via the policy gradient method was also studied in [FGKM18].

The closest work to ours is that of [AYBK14] and [CHK$^+$18], aimed at controlling LDS with adversarial loss functions. The authors in [AYS11] obtain a $O(\log^2 T)$ regret algorithm for changing quadratic costs (with a fixed hessian), but for dynamical systems that are noise-free. In contrast, our results apply to the full (noisy) LDS setting, which presents the main challenges as discussed before. Cohen et al. [CHK$^+$18] consider changing quadratic costs with stochastic noise to achieve a $O(\sqrt{T})$ regret bound.

We make extensive use of techniques from online learning [CBL06, SS$^+$12, Haz16]. Of particular interest to our study is the setting of online learning with memory [AHM15]. We also build upon the recent control work of [ABH$^+$19], who use online learning techniques and convex relaxation to

obtain provable bounds for LDS with adversarial perturbations.

3.2 Problem setting

We consider a linear dynamical system as defined in Equation 3.1.1 with costs $c_t(x_t, u_t)$, where c_t is strongly convex. In this chapter we assume that the noise w_t is a random variable generated independently at every time step. For any algorithm $\mathcal{A}$, we attribute a cost defined as

$$J_T(\mathcal{A}) = \mathbb{E}_{\{w_t\}}\left[\sum_{t=1}^{T} c_t(x_t, u_t)\right],$$

where $x_{t+1} = Ax_t + Bu_t + w_t$, $u_t = \mathcal{A}(x_1, \dots x_t)$ and $\mathbb{E}_{\{w_t\}}$ represents the expectation over the entire noise sequence. For the rest of the chapter we will drop the subscript $\{w_t\}$ from the expectation as it will be the only source of randomness. Overloading notation, we shall use $J_T(K)$ to denote the cost of a linear controller K which chooses the action as $u_t = -Kx_t$.

Assumptions. In the chapter we assume that $x_1 = 0$ [1], as well as the following conditions.

Assumption 3.2.1. *We assume that $\|B\| \leq \kappa_B$. Furthermore, the perturbation introduced per time step is bounded, i.i.d, and zero-mean with a lower bounded covariance i.e.*

$$\forall t\ w_t \sim \mathcal{D}_w, \mathbb{E}[w_t] = 0, \mathbb{E}[w_t w_t^\top] \succeq \sigma^2 I \text{ and } \|w_t\| \leq W$$

This may be adapted to the case of sub-gaussian noise by conditioning on the event that none of the noise vectors are ever large. Such adaptation introduces a multiplicative $\log(T)$ factor in the regret.

Assumption 3.2.2. *The costs $c_t(x, u)$ are α-strongly convex. Wehnever $\|x\|, \|u\| \leq D$, it holds that*

$$\|\nabla_x c_t(x, u)\|, \|\nabla_u c_t(x, u)\| \leq GD.$$

The class of linear controllers we work with are defined as follows; see the subsection below for a detailed note.

Definition 3.2.3 (Diagonal Strong Stability)**.** *Given a dynamics (A, B), a linear controller K is (κ, γ)-diagonal strongly stable for real numbers $\kappa \geq 1, \gamma < 1$, if there exists a complex diagonal matrix L and a non-singular complex matrix Q, such that $A - BK = QLQ^{-1}$ with the following being true:*

[1] This is only for convenience of presentation. The case with a bounded x_1 can be handled similarly.

1. *The spectral norm of L is strictly smaller than one, i.e., $\|L\| \leq 1 - \gamma$.*

2. *The controller and transforming matrices are bounded, i.e., $\|K\| \leq \kappa$ and $\|Q\|, \|Q^{-1}\| \leq \kappa$.*

Regret Formulation. Let $\mathcal{K} = \{K : K \text{ is } (\kappa, \gamma)\text{-diagonal strongly stable}\}$. For an algorithm $\mathcal{A}$, the notion of regret we consider is *pseudo-regret*, i.e. the sub-optimality of its cost with respect to the cost for the best linear controller i.e.,

$$\texttt{Regret} = J_T(\mathcal{A}) - \min_{K \in \mathcal{K}} J_T(K).$$

3.2.1 Discussion on diagonal strong stability

Classically, a controller K is stabilizing [Ste94b] if the spectral radius of $A - BK \leq 1 - \delta$. The notion of strong stability was introduced by [CHK$^+$18] – being the same as Definition 3.2.3, but not requiring L to be diagonal. Both strong stability and diagonal strong stability are quantitative measures of the classical notion of stabilizing controllers that permit a discussion on non-asymptotic regret bounds. We note that an analogous notion for quantification of open-loop stability appears in the work of [HLS$^+$18].

On the generality of the diagonal strong stability notion, the following comment may be made: while not all matrices are complex diagonalizable, an exhaustive characterization of $m \times m$ complex diagonal matrices is the existence of m linearly independent eigenvectors; for the later, it suffices, but is not necessary, that a matrix has m distinct eigenvalues (See [Str]). It may be observed that almost all matrices admit distinct eigenvalues, and hence, are complex diagonalizable insofar the complement set admits a zero-measure. By this discussion, *almost all* stabilizing controllers are diagonal strongly stable for some κ, γ. The astute reader may note the departure here from the more general notion – strongly stability – in that *all* stabilizing controllers are strongly stable for some choice of parameters.

3.3 Preliminaries

Notation. We reserve the letters x, y for states and u, v for actions. We denote by d_x, d_u to be the dimension of the state and control space respectively. Let $d = \max(d_x, d_u)$. We reserve capital letters A, B, K, M for matrices associated with the system and the policy. Other capital letters are reserved for universal constants. We use the shorthand $M_{i:j}$ to denote a subsequence $\{M_i, \ldots, M_j\}$. For any matrix U, define U_{vec} to be a flattening of the matrix where we stack the rows vertically upon each

other. For a collection of matrices $M = \{M^{[i]}\}$, let M_{vec} be the flattening defined by stacking the flattenings of $M^{[i]}$ upon each other. We use $\|x\|_U^2 = x^\top U x$ to denote the matrix induced norm. The rest of this section provides a recap of the relevant definitions and concepts introduced in [ABH+19].

3.3.1 Reference policy class

For the rest of the chapter, we fix a (κ, γ)-diagonally strongly stable matrix $\mathbb{K}$ (The bold notation is to stress that we treat this matrix as fixed and not a parameter). Note that this can be any such matrix and it can be computed via a semi-definite feasibility program [CHK+18] given the knowledge of the dynamics, before the start of the game. We work with following the class of policies.

Definition 3.3.1 (Disturbance-Action Policy)**.** *A disturbance-action policy $M = (M^{[0]}, \ldots, M^{[H-1]})$, for horizon $H \geq 1$ is defined as the policy which at every time t, chooses the recommended action u_t at a state x_t, defined* [2] *as*

$$u_t(M) \triangleq -\mathbb{K}x_t + \sum_{i=1}^{H} M^{[i-1]} w_{t-i}.$$

For notational convenience, here it may be considered that $w_i = 0$ for all $i < 0$.

The policy applies a linear transformation to the disturbances observed in the past H steps. Since (x, u) is a linear function of the disturbances in the past under a linear controller K, formulating the policy this way can be seen as a relaxation of the class of linear policies. Note that $\mathbb{K}$ is a fixed matrix and is not part of the parameterization of the policy. As was established in the previous chapter (also)[ABH+19]), with appropriate choice of parameters, superimposing such a $\mathbb{K}$, allows the policy class to approximate any linear policy in terms of the cost with a finite horizon parameter H.

We refer to the policy played at time t as $M_t = \{M_t^{[i]}\}$ where the subscript t refers to the time index and the superscript $[i-1]$ refers to the action of M_t on w_{t-i}. Note that such a policy can be executed because w_{t-1} is perfectly determined on the specification of x_t as $w_{t-1} = x_t - Ax_{t-1} - Bu_{t-1}$.

3.3.2 Evolution of state

This section describes the evolution of the state of the linear dynamical system under a non-stationary policy composed of a sequence of T policies, where at each time the policy is specified by $M_t = (M_t^{[0]}, \ldots, M_t^{[H-1]})$. We will use $M_{0:T-1}$ to denote such a non-stationary policy. The following definitions ease the burden of notation.

[2] x_t is completely determined given $w_0 \ldots w_{t-1}$. Hence, the use of x_t only serves to ease the burden of presentation.

1. Define $\tilde{A} = A - B\mathbb{K}$. $\tilde{A}$ shall be helpful in describing the evolution of state starting from a non-zero state in the absence of disturbances.

2. For any sequence of matrices $M_{0:H}$, define Ψ_i as a linear function that describes the effect of w_{t-i} on the state x_t, formally defined below.

Definition 3.3.2. *For any sequence of matrices $M_{0:H}$, define the disturbance-state transfer matrix Ψ_i for $i \in \{0, 1, \dots H\}$, to be a function with $h+1$ inputs defined as*

$$\Psi_i(M_{0:H}) \triangleq \tilde{A}^i \mathbf{1}_{i \le H} + \sum_{j=0}^{H} \tilde{A}^j B M_{H-j}^{[i-j-1]} \mathbf{1}_{i-j \in [1,H]}.$$

It will be important to note that ψ_i is a **linear** function of its argument.

3.3.3 Surrogate state and surrogate cost

This section introduces a couple of definitions required to describe our main algorithm. In essence they describe a notion of state, its derivative and the expected cost if the system evolved solely under the past H steps of a non-stationary policy.

Definition 3.3.3 (Surrogate State & Surrogate Action)**.** *Given a sequence of matrices $M_{0:H+1}$ and $2H$ independent invocations of the random variable w given by $\{w_j \sim \mathcal{D}_w\}_{j=0}^{2H-1}$, define the following random variables denoting the surrogate state and the surrogate action:*

$$y(M_{0:H}) = \sum_{i=0}^{2H} \Psi_i(M_{0:H}) w_{2H-i-i}, \qquad v(M_{0:H+1}) = -\mathbb{K} y(M_{0:H}) + \sum_{i=1}^{H} M_{H+1}^{[i-1]} w_{2H-i}.$$

When M is the same across all arguments we compress the notation to $y(M)$ and $v(M)$ respectively.

Definition 3.3.4 (Surrogate Cost)**.** *Define the surrogate cost function f_t to be the cost associated with the surrogate state-action pair defined above, i.e., $f_t(M_{0:H+1}) = \mathbb{E}\left[c_t(y(M_{0:H}), v(M_{0:H+1}))\right]$. When M is the same across all arguments we compress the notation to $f_t(M)$.*

Definition 3.3.5 (Jacobian)**.** *Let $z(M) = \begin{bmatrix} y(M) \\ v(M) \end{bmatrix}$. Since $y(M), v(M)$ are random linear functions of M, $z(M)$ can be reparameterized as $z(M) = JM_{vec} = \begin{bmatrix} J_y \\ J_v \end{bmatrix} M_{vec}$, where J is a random matrix, which derives its randomness from the random perturbations w_i.*

Algorithm 3 Online Control Algorithm

Input: Step size schedule η_t, Parameters $\kappa_B, \kappa, \gamma, T$.
Define $H = \gamma^{-1}\log(T\kappa^2)$
Define $\mathcal{M} = \{M = \{M^{[0]} \dots M^{[H-1]}\} : \|M^{[i-1]}\| \leq \kappa^3\kappa_B(1-\gamma)^i\}$.
Initialize $M_0 \in \mathcal{M}$ arbitrarily.
for $t = 0, \dots, T-1$ **do**
    Choose the action:

$$u_t = -\mathbb{K}x_t + \sum_{i=1}^{H} M_t^{[i-1]} w_{t-i}.$$

    Observe the new state x_{t+1} and record $w_t = x_{t+1} - Ax_t - Bu_t$.
    Online Gradient Update:

$$M_{t+1} = \Pi_{\mathcal{M}}(M_t - \eta_t \nabla f_t(M_t))$$

    Online Natural Gradient Update:

$$M_{vec,t+1} = \Pi_{\mathcal{M}}(M_{vec,t} - \eta_t(\mathbb{E}[J^T J])^{-1}\nabla_{M_{vec,t}} f_t(M_t))$$

end for

3.3.4 OCO with memory

We now describe the setting of online convex optimization with memory introduced in [AHM15]. In this setting, at every step t, an online player chooses some point $x_t \in \mathcal{K} \subset \mathbb{R}^d$, a loss function $f_t : \mathcal{K}^{H+1} \mapsto \mathbb{R}$ is then revealed, and the learner suffers a loss of $f_t(x_{t-H:t})$. We assume coordinate-wise Lipschitz regularity on f_t of the form such that, for any $j \in \{0, \dots, H\}$, for any $x_{0:H}, \tilde{x}_j \in \mathcal{K}$,

$$|f_t(x_{0:j-1}, x_j, x_{j+1:H}) - f_t(x_{0:j-1}, \tilde{x}_j, x_{j+1:H})| \leq L\|x_j - \tilde{x}_j\|. \tag{3.3.1}$$

In addition, we define $f_t(x) = f_t(x, \dots, x)$, and we let

$$G_f = \sup_{t\in\{0,\dots,T\}, x\in\mathcal{K}} \|\nabla f_t(x)\|, \quad D = \sup_{x,y\in\mathcal{K}} \|x - y\|. \tag{3.3.2}$$

The resulting goal is to minimize the *policy regret* [ADT12], which is defined as

$$\texttt{PolicyRegret} = \sum_{t=H}^{T} f_t(x_{t-H:t}) - \min_{x\in\mathcal{K}} \sum_{t=H}^{T} f_t(x).$$

3.4 Algorithms & results

The two variants of our method are spelled out in Algorithm 3. Theorems 3.4.1 and 3.4.3 provide the main guarantees for the two algorithms.

Online Gradient Update

Theorem 3.4.1 (Online Gradient Update)**.** *Suppose Algorithm 3 (Online Gradient Update) is executed with $\mathbb{K}$ being any (κ, γ)-diagonal strongly stable matrix and $\eta_t = \Theta\left(\alpha\sigma^2 t\right)^{-1}$, on an LDS satisfying Assumption 3.2.1 with control costs satisfying Assumption 3.2.2. Then, it holds true that*

$$J_T(\mathcal{A}) - \min_{K\in\mathcal{K}} J_T(K) \leq \tilde{O}\left(\frac{G^2W^4}{\alpha\sigma^2}\log^7(T)\right).$$

The above result leverages the following lemma which shows that the function $f_t(\cdot)$ is strongly convex with respect to its argument M. Note that strong convexity of the cost functions c_t over the state-action space does not by itself imply the strong convexity of the surrogate cost f_t over the space of controllers M. This is because, in the surrogate cost f_t, c_t is applied to $y(M), v(M)$ which themselves are linear functions of M; the linear map M is necessarily column-rank-deficient. To observe this, note that M maps from a space of dimensionality $H \times \dim(x) \times \dim(u)$ to that of $\dim(x) + \dim(u)$. The next theorem, which forms the core of our analysis, shows that this is not the case using the inherent stochastic nature of the dynamical system.

Lemma 3.4.2. *If the cost functions $c_t(\cdot,\cdot)$ are α-strongly convex, $\mathbb{K}$ is (κ,γ)-diagonal strongly stable and Assumption 3.2.1 is met, then $f_t(M)$ is λ-strongly convex with respect to M where*

$$\lambda = \frac{\alpha\sigma^2\gamma^2}{36\kappa^{10}}$$

We present the proof for simple cases in Section 3.6, deferring the general proof to Section 3.11.

Online Natural Gradient Update

Theorem 3.4.3 (Online Natural Gradient Update)**.** *Suppose Algorithm 3 (Online Natural Gradient Update) is executed with $\eta_t = \Theta\left(\alpha t\right)^{-1}$, on an LDS satisfying Assumptions 3.2.1 and with control costs satisfying Assumption 3.2.2. Then, it holds true that*

$$J_T(\mathcal{A}) - \min_{K\in\mathcal{K}} J_T(K) \leq \tilde{O}\left(\frac{GW^2}{\alpha\mu}\log^7(T)\right) \quad \textit{where} \quad \mu^{-1} \triangleq \max_{M\in\mathcal{M}} \|(\mathbb{E}[J^T J])^{-1}\nabla_{M_{vec}} f_t(M)\|.$$

In Theorem 3.4.3, the regret guarantee depends on an instance-dependent parameter μ, which is a measure of hardness of the problem. First, we note that the proof of Lemma 3.4.2 establishes that the Gram matrix of the Jacobian (Defintion 3.3.5) is strictly positive definite and hence we recover the log regret guarantee achieved by the Online Gradient Descent Update, with the constants preserved.

Corollary 3.4.4. *In addition to the assumptions in Theorem 3.4.3, if $\mathbb{K}$ is a (κ, γ)-diagonal strongly stable matrix, then for the natural gradient update*

$$J_T(\mathcal{A}) - \min_{K \in \mathcal{K}} J_T(K) \leq \tilde{O}\left(\frac{G^2 W^4}{\alpha \sigma^2} \log^7(T)\right),$$

Proof. The conclusion follows from Lemma 3.5.2 and Lemma 3.6.1 which is the core component in the proof of Lemma 3.4.2 showing that $\mathbb{E}[J^T J] \geq \frac{\gamma^2 \sigma^2}{36 \kappa^{10}} \cdot \mathbb{I}$. □

Secondly, we note that, being instance-dependent, the guarantee the Natural Gradient update offers can potentially be stronger than that of the Online Gradient method. A case in point is the following corollary involving spherically symmetric quadratic costs, in which case the Natural Gradient update yields a regret guarantee under demonstrably more general conditions, in that the bound does not depend on the minimum eigenvalue of the covariance of the disturbances σ^2, unlike OGD.

Corollary 3.4.5. *Under the assumptions on Theorem 3.4.3, if the cost functions are of the form $c_t(x, u) = r_t(\|x\|^2 + \|u\|^2)$, where $r_t \in [\alpha, \beta]$ is an adversarially chosen sequence of numbers and $\mathbb{K}$ is chosen to be a (κ, γ)-diagonal strongly stable matrix, then the natural gradient update guarantees*

$$J_T(\mathcal{A}) - \min_{K \in \mathcal{K}} J_T(K) \leq \tilde{O}\left(\frac{\beta^2 W^2}{\alpha} \log^7(T)\right).$$

Proof. Note $\|\nabla_{M_{\text{vec}}} f_t(M)\|_{(\mathbb{E}[J^T J])^{-2}} = \|\mathbb{E}[J^T (r_t \cdot I) J M_{\text{vec}}]\|_{(\mathbb{E}[J^T J])^{-2}} \leq \beta \|M_{\text{vec}}\|$. □

3.5 Regret analysis

The next section is a condensation of the results from the previous chapter which we present in this form to highlight the reduction to OCO with memory. The simplified stochastic setting also permits a more compact notaion in some instances.

3.5.1 Reduction to low regret with memory

The next lemma shows that achieving low policy regret on the memory based function f_t is sufficient to ensure low regret on the overall dynamical system. Define,

$$\mathcal{M} \triangleq \{M = \{M^{[0]} \dots M^{[H-1]}\} : \|M^{[i-1]}\| \leq \kappa^3 \kappa_B (1-\gamma)^i\}.$$

Lemma 3.5.1. *Let the dynamical system satisfy Assumption 3.2.1 and let $\mathbb{K}$ be any (κ, γ)-diagonal strongly stable matrix. Consider a sequence of loss functions $c_t(x, u)$ satisfying Assumption 3.2.2 and*

a sequence of policies $M_0 \dots M_T$ satisfying

$$\texttt{PolicyRegret} = \sum_{t=0}^{T} f_t(M_{t-H-1:t}) - \min_{M \in \mathcal{M}} \sum_{t=0}^{T} f_t(M) \leq R(T)$$

for some function $R(T)$ and f_t as defined in Definition 3.3.4. Let A be an online algorithm that plays the non-stationary controller sequence $\{M_0, \dots M_T\}$. Then as long as H is chosen to be larger than $\gamma^{-1} \log(T\kappa^2)$ we have that

$$J(A) - \min_{K^* \in \mathcal{K}} J(K^*) \leq R(T) + O(GW^2 \log(T)),$$

Here $O(\cdot)$, $\Theta(\cdot)$ contain polynomial factors in $\gamma^{-1}, \kappa_B, \kappa, d$.

Lemma 3.5.2. *The function f_t as defined in Definition 3.3.4 is coordinate-wise L-lipschitz and the norm of the gradient is bounded by G_f, where*

$$L = \frac{2DGW\kappa_B\kappa^3}{\gamma}, \;\; G_f \leq GDWHd\left(H + \frac{2\kappa_B\kappa^3}{\gamma}\right)$$

$$\text{where } D \triangleq \frac{W\kappa^2(1 + H\kappa_B^2\kappa^3)}{\gamma(1 - \kappa^2(1-\gamma)^{H+1})} + \frac{\kappa_B\kappa^3 W}{\gamma}.$$

The proof of this lemma is identical to the analogous lemma in [ABH+19] and hence is omitted.

3.5.2 Analysis for online gradient descent

In the setting of Online Convex Optimization with Memory, as shown by [AHM15], by running a memory-based OGD, we can bound the policy regret by the following theorem.

Theorem 3.5.3. *Consider the OCO with memory setting defined in Section 3.3.4. Let $\{f_t\}_{t=H}^T$ be Lipschitz loss functions with memory such that $f_t(x)$ are λ-strongly convex, and let L and G_f be as defined in equation 3.3.1 and equation 3.3.2. Then, there exists an algorithm which generates a sequence $\{x_t\}_{t=0}^T$ such*

$$\sum_{t=H}^{T} f_t(x_{t-H:t}) - \min_{x \in \mathcal{K}} \sum_{t=H}^{T} \tilde{f}_t(x) \leq \frac{G_f^2 + LH^2G_f}{\lambda}(1 + \log(T)).$$

Proof of Theorem 3.4.1. Setting $H = \gamma^{-1}\log(T\kappa^2)$, Theorem 3.5.3, in conjunction with Lemma 3.5.2, implies that policy regret is bounded by $\tilde{O}\left(\frac{G^2W^4H^6}{\alpha\sigma^2}\log T\right)$. An invocation of Lemma 3.5.1 now suffices to conclude the proof of the claim. □

3.5.3 Analysis for online natural gradient descent

Consider structured loss functions of the form $f_t(M_{0:H+1}) = \mathbb{E}[c_t(z)]$, where $z = \sum_{i=0}^{H+1} J_i[M_i]_{\text{vec}}$. J_i is a random matrix, and c_t's are adversarially chosen strongly convex loss functions. In a similar vein, define $f_t(M)$ to be the specialization of f_t when input the same argument, i.e. M, $H+1$ times. Define $J = \sum_{i=0}^{H+1} J_i$.

Theorem 3.5.4. *In the setting desribed in this subsection, let c_t be α-strongly convex, and f_T be such that it satisfies equation 3.3.1 with constant L, and $G_f = \max_{M\in\mathcal{M}} \|(\mathbb{E}[J^T J])^{-1}\nabla_{M_{vec}} f_t(M)\|$. Then, the online natural gradient update generates a sequence $\{M_t\}_{t=0}^T$ such that*

$$\sum_{t=H}^{T} f_t(M_{t-H:t}) - \min_{M\in\mathcal{M}} \sum_{t=H}^{T} \tilde{f}_t(M) \leq \frac{\max_{M\in\mathcal{M}} \|\nabla_{M_{vec}} f_t(M)\|^2_{(\mathbb{E}[J^T J])^{-1}} + LH^2 G_f}{\alpha}(1+\log(T)).$$

Proof of Theorem 3.4.3. First observe that $\|\nabla_{M_{\text{vec}}} f_t(M)\|^2_{(\mathbb{E}[J^T J])^{-1}} \leq \mu^{-1}\|\nabla_{M_{\text{vec}}} f_t(M)\|$. Setting $H = \gamma^{-1}\log(T\kappa^2)$, Theorem 3.5.4, in conjunction with Lemma 3.5.2, imply the stated bound on policy regret. An invocation of Lemma 3.5.1 suffices to conclude the proof of the claim. □

3.6 Proof of strong convexity in simple cases

We will need some definitions and preliminaries that are outlined below. By definition we have that $f_t(M) = \mathbb{E}[c_t(y_t(M), v_t(M))]$. Since we know that c_t is strongly convex we have that

$$\nabla^2 f_t(M) = \mathbb{E}_{\{w_k\}_{k=0}^{2H-1}}[\nabla^2 c_t(y(M), v(M))] \succeq \alpha \mathbb{E}_{\{w_k\}_{k=0}^{2H-1}}[J_y^\top J_y + J_v^\top J_v].$$

J_y, J_v are random matrices dependent on the noise $\{w_k\}_{k=0}^{2H-1}$. The next lemma implies Lemma 3.4.2.

Lemma 3.6.1. *If Assumption 3.2.1 is satisfied and $\mathbb{K}$ is chosen to be a (κ,γ)-diagonal strongly stable matrix, then the following holds,*

$$\mathbb{E}_{\{w_k\}_{k=0}^{2H-1}}[J_y^\top J_y + J_v^\top J_v] \succeq \frac{\gamma^2\sigma^2}{36\kappa^{10}} \cdot \mathbb{I}.$$

To analyze J_y, J_v, we will need to rearrange the definition of $y(M)$ to make the dependence on each individual $M^{[i]}$ explicit. To this end consider the following definition for all $k \in [H+1]$.

$$\tilde{v}_k(M) \triangleq \sum_{i=1}^{H} M^{[i-1]} w_{2H-i-k}$$

Under this definition it follows that

$$y(M) = \sum_{k=1}^{H} (A - B\mathbb{K})^{k-1} B\tilde{v}_k(M) + \sum_{k=1}^{H} (A - B\mathbb{K})^{k-1} w_{2H-k}$$

$$v(M) = -\mathbb{K}y(M) + \tilde{v}_0(M)$$

From the above definitions, (J_y, J_v) may be characterized in terms of the Jacobian of $\tilde{v}_k$ with respect to M, which we define for the rest of the section as $J_{\tilde{v}_k}$. Defining M_{vec} as the stacking of rows of each $M^{[i]}$ vertically, i.e. stacking the columns of $(M^{[i]})^\top$, it can be observed that for all k,

$$J_{\tilde{v}_k} = \frac{\partial \tilde{v}_k(M)}{\partial M} = \begin{bmatrix} I_{d_u} \otimes w_{2H-k-1}^\top & I_{d_u} \otimes w_{2H-k-2}^\top & \dots & I_{d_u} \otimes w_{H-k}^\top \end{bmatrix}$$

where d_u is the dimension of the controls. We are now ready to analyze the two simpler cases. Further on in the section we drop the subscripts $\{w_k\}_{k=0}^{2H-1}$ from the expectations for brevity.

3.6.1 Stable linear systems

In this section we assume that $\mathbb{K} = 0$ is a (κ, γ)-diagonal strongly stable policy for (A, B). Be definition, we have $v(M) = \tilde{v}_0(M)$. One may conclude the proof with the following observation.

$$\mathbb{E}[J_y^\top J_y + J_v^\top J_v] \succeq \mathbb{E}[J_v^\top J_v] = \mathbb{E}[J_{\tilde{v}_0}^\top J_{\tilde{v}_0}] = I_{d_u} \otimes \Sigma \succeq \sigma^2 \mathbb{I}.$$

3.6.2 1-dimensional stabilizable case

Here, we specialize Lemma 3.4.2 to one-dimensional state and one-dimensional control. This case highlights the difficulty caused in the proof due to a choosing a non-zero $\mathbb{K}$ and presents the main ideas of the proof in a simplified notation.

Note that in the one dimensional case, the policy given by $M = \{M^{[i]}\}_{i=0}^{H-1}$ is an H dimensional vector with $M^{[i]}$ being a scalar. Furthermore $y(M), v(M), \tilde{v}_k(M)$ are scalars and hence their Jacobians $J_y, J_v, J_{\tilde{v}_k}$ with respect to M are $1 \times H$ vectors. In particular we have that,

$$J_{\tilde{v}_k} = \frac{\partial \tilde{v}_k(M)}{\partial M} = \begin{bmatrix} w_{2H-k-1} & w_{2H-k-2} & \dots & w_{H-k} \end{bmatrix}$$

Therefore using the fact that $E[w_i w_j] = 0$ for $i \neq j$ and $\mathbb{E}[w_i^2] = \sigma^2$, it can be observed that for any k_1, k_2, we have that

$$\mathbb{E}[J_{v_{\tilde{k}_1}}^\top J_{v_{\tilde{k}_2}}] = \mathcal{T}_{k_1 - k_2} \cdot \sigma^2 \tag{3.6.1}$$

where $\mathcal{T}_m$ is defined as an $H \times H$ matrix with $[\mathcal{T}_m]_{ij} = 1$ if and only if $i - j = m$ and 0 otherwise. This in particular immediately gives us that,

$$\mathbb{E}[J_y^\top J_y] = \underbrace{\left(\sum_{k_1=1}^{H} \sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \cdot (A - B\mathbb{K})^{k_1-1+k_2-1} \right)}_{\triangleq \mathbb{G}} \cdot B^2 \cdot \sigma^2 \tag{3.6.2}$$

$$\mathbb{E}[J_{\tilde{v}_0}^\top J_y] = \underbrace{\left(\sum_{k=1}^{H} \mathcal{T}_{-k} (A - B\mathbb{K})^{k-1} \right)}_{\triangleq \mathbb{Y}} \cdot B \cdot \sigma^2 \tag{3.6.3}$$

First, we prove a few spectral properties of the matrices $\mathbb{G}$ and $\mathbb{Y}$ defined above. From Gershgorin's circle theorem, and the fact that $\mathbb{K}$ is (κ, γ)-diagonal strongly stable, we have

$$\|\mathbb{Y} + \mathbb{Y}^\top\| \leq \| \sum_{k=1}^{H} (\mathcal{T}_{-k} + \mathcal{T}_k)(A - B\mathbb{K})^{k-1} \| \leq 2\gamma^{-1} \tag{3.6.4}$$

The spectral properties of $\mathbb{G}$ summarized in the lemma below form the core of our analysis.

Lemma 3.6.2. *$\mathbb{G}$ is a symmetric positive definite matrix. In particular $\mathbb{G} \succeq \frac{1}{4} \cdot I$.*

Now consider the statements which follow by the respective definitions.

$$\begin{aligned} \mathbb{E}[J_v^\top J_v] &= \mathbb{K}^2 \cdot \mathbb{E}[J_y^\top J_y] - \mathbb{K} \cdot \mathbb{E}[J_y^\top J_{\tilde{v}_0}] - \mathbb{K} \cdot \mathbb{E}[J_{\tilde{v}_0}^\top J_y] + \mathbb{E}[J_{\tilde{v}_0}^\top J_{\tilde{v}_0}] \\ &= \sigma^2 \cdot \underbrace{\left(B^2\mathbb{K}^2 \cdot \mathbb{G} - B\mathbb{K} \cdot (\mathbb{Y} + \mathbb{Y}^\top) + I \right)}_{\triangleq \mathbb{F}}. \end{aligned}$$

Now $\mathbb{F} \succeq 0$. We finish the proof by considering two cases. The first case is when $3|B|\gamma^{-1}\kappa \geq 1$. Noting $\kappa \geq 1$, in this case Lemma 3.6.2 immediately implies that

$$m^\top \left(\mathbb{F} + B^2 \cdot \mathbb{G} \right) m \geq m^\top \left(B^2 \cdot \mathbb{G} \right) m \geq \frac{\frac{1}{4}\|m\|^2}{9\gamma^{-2}\kappa^2} \geq \frac{\gamma^2 \|m\|^2}{36\kappa^{10}},$$

In the second case (when $3|B|\gamma^{-1}\kappa \leq 1$), equation 3.6.4 implies that

$$m^\top \left(\mathbb{F} + B^2 \cdot \mathbb{G} \right) m \geq m^\top \left(I - B\mathbb{K} \cdot (\mathbb{Y} + \mathbb{Y}^\top) \right) m \geq (1/3)\|m\|^2 \geq \frac{\gamma^2 \|m\|^2}{36\kappa^{10}}.$$

Proof of Lemma 3.6.2

Recall $\mathcal{T}_m$ is defined as an $H \times H$ matrix with $[\mathcal{T}_m]_{ij} = 1$ if and only if $i - j = m$ and 0 otherwise. Define the following matrix for any complex number $|\psi| < 1$.

$$\mathbb{G}(\psi) = \sum_{k_1=1}^{H} \sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \left(\psi^\dagger\right)^{k_1-1} \psi^{k_2-1}$$

Note that $\mathbb{G}$ in Lemma 3.6.2 is equal to $\mathbb{G}(A - B\mathbb{K})$. The following lemma, proven in Section 3.10, provides a lower bound on the spectrum of the matrix $\mathbb{G}(\psi)$. The lemma presents the proof of a more general case (ϕ is complex) that aids the multi-dimensional case. A special case when $\phi = 1$ was proven in [GKM18], and we follow a similar approach relying on the inverse.

Lemma 3.6.3. *Let ψ be a complex number such that $|\psi| \leq 1$. We have that $\mathbb{G}(\psi) \succeq (1/4) \cdot I_H$.*

3.7 Proof of the reductions to OCO with memory

Since the proof borrows heavily from the definitions introduced by the previous chapter, we restate those definitions here for convenience, while taking the oppurtunity to simplify the notation for the stochastic noise setting.

1. Let $x_t^K(M_{0:t-1})$ is the state attained by the system upon execution of a non-stationary policy $\pi(M_{0:t-1}, K)$. We similarly define $u_t^K(M_{0:t-1})$ to be the action executed at time t. If the same policy M is used across all time steps, we compress the notation to $x_t^K(M), u_t^K(M)$. Note that $x_t^K(0), u_t^K(0)$ refers to running the linear policy K.

2. $\Psi_{t,i}^{K,h}(M_{t-h:t})$ is a transfer matrix that describes the effect of w_{t-i} with respect to the past $h+1$ policies on the state x_{t+1}, formally defined below. When M is the same across all arguments we compress the notation to $\Psi_{t,i}^{K,h}(M)$.

Definition 3.7.1. *For any $t, h \leq t, i \leq H + h$, define the disturbance-state transfer matrix $\Psi_{t,i}^{K,h}$ to be a function with $h + 1$ inputs defined as*

$$\Psi_{t,i}^{K,h}(M_{t-h:t}) = \tilde{A}_K^i \mathbf{1}_{i \leq h} + \sum_{j=0}^{h} \tilde{A}_K^j B M_{t-j}^{[i-j-1]} \mathbf{1}_{i-j \in [1,H]}.$$

Definition 3.7.2 (Surrogate State & Surrogate Action)**.** *Define,*

$$y_{t+1}^K(M_{t-H:t}) = \sum_{i=0}^{2H} \Psi_{t,i}^{K,H}(M_{t-H:t}) w_{t-i},$$

$$v_{t+1}^K(M_{t-H:t+1}) = -K y_{t+1}^K(M_{t-H:t}) + \sum_{i=1}^{H} M_{t+1}^{[i-1]} w_{t+1-i}.$$

When M is the same across all arguments we compress the notation to $y_{t+1}^K(M), v_{t+1}^K(M)$.

Definition 3.7.3 (Surrogate Cost)**.** *Define the surrogate cost function f_t to be the cost associated with the surrogate state and surrogate action, i.e.,*

$$f_t(M_{t-H-1:t}) = \mathbb{E}\left[c_t(y_t^K(M_{t-H-1:t-1}), v_t^K(M_{t-H-1:t}))\right].$$

When M is the same across all arguments we compress the notation to $f_t(M)$.

Note that this definition coincides exactly with Definition 3.3.4 in the main text.

In this paragraph we state some lemmas and theorems which were proved in [ABH+19]. Due to consistency of definitions the proofs of these are omitted and can be found in [ABH+19].

Lemma 3.7.4 (Sufficiency)**.** *For any two (κ, γ)-diagonal strongly stable matrices K^*, K, there exists $M_* = (M_*^{[0]}, \ldots, M_*^{[H-1]}) \in \mathcal{M}$ defined as*

$$M_*^{[i]} = (K - K^*)(A - BK^*)^i$$

such that

$$\sum_{t=0}^{T} \left(c_t(x_t^K(M_*), u_t^K(M_*)) - c_t(x_t^{K^*}(0), u_t^{K^*}(0)) \right) \leq T \cdot \frac{2GDWH\kappa_B^2\kappa^5(1-\gamma)^H}{\gamma}.$$

Theorem 3.7.5. *For any (κ, γ)-diagonal strongly stable K, any $\tau > 0$, and any sequence of policies $M_1 \ldots M_T$ satisfying $\|M_t^{[i]}\| \leq \tau(1-\gamma)^i$, if the perturbations are bounded by W, we have that*

$$\sum_{t=1}^{T} f_t(M_{t-H-1:t}) - \sum_{t=1}^{T} c_t(x_t^K(M_{0:t-1}), u_t^K(M_{0:t})) \leq 2TGD^2\kappa^3(1-\gamma)^{H+1},$$

where

$$D \triangleq \frac{W\kappa^3(1 + H\kappa_B\tau)}{\gamma(1 - \kappa^2(1-\gamma)^{H+1})} + \frac{\tau W}{\gamma}.$$

Proof of Lemma 3.5.1. Let D be defined as

$$D \triangleq \frac{W\kappa^3(1+H\kappa_B\tau)}{\gamma(1-\kappa^2(1-\gamma)^{H+1})} + \frac{\kappa_B\kappa^3 W}{\gamma}.$$

Let K^* be the optimal linear policy in hindsight. By definition K^* is a (κ,γ)-diagonal strongly stable matrix. Using Lemma 3.7.4 and Theorem 3.7.5, we have that

$$\begin{aligned}
&\min_{M_*\in\mathcal{M}}\left(\sum_{t=0}^{T} f_t(M_*)\right) - \sum_{t=0}^{T} c_t(x_t^{K^*}(0), u_t^{K^*}(0)) \\
&\leq \min_{M_*\in\mathcal{M}}\left(\sum_{t=0}^{T} c_t(x_t^{K}(M_*), u_t^{K}(M_*))\right) - \sum_{t=0}^{T} c_t(x_t^{K^*}(0), u_t^{K^*}(0)) + 2TGD^2\kappa^3(1-\gamma)^{H+1} \\
&\leq 2TGD(1-\gamma)^{H+1}\left(\frac{WH\kappa_B^2\kappa^5}{\gamma} + D\kappa^3\right). \qquad (3.7.1)
\end{aligned}$$

Note that by definition of $\mathcal{M}$, we have that

$$\forall t\in[T], \forall i\in[H] \quad \|M_t^{[i]}\| \leq \kappa_B\kappa^3(1-\gamma)^i.$$

Using Theorem 3.7.5 we have that

$$\sum_{t=0}^{T} c_t(x_t^K(M_{0:t-1}), u_t^K(M_{0:t-1})) - \sum_{t=0}^{T} f_t(M_{t-H-1:t}) \leq 2TGD^2\kappa^3(1-\gamma)^{H+1}. \qquad (3.7.2)$$

Summing up equation 3.7.1 and equation 3.7.2 and using the condition that $H \geq \frac{1}{\gamma}\log(T\kappa^2)$, we get the result. □

3.8 Policy regret for online gradient descent

Proof of Theorem 3.5.3. By the standard OGD strong convexity analysis, if $\eta_t = (\lambda\cdot(t-H))^{-1}$, we have that

$$\sum_{t=H}^{T}\tilde{f}_t(x_t) - \min_{x\in\mathcal{K}}\sum_{t=H}^{T}\tilde{f}_t(x) \leq \frac{G^2}{2\lambda}(1+\log(T)).$$

In addition, we know by equation 3.3.1 that, for any $t \geq H$,

$$\begin{aligned}
|f_t(x_{t-H},\ldots,x_t) - f_t(x_t,\ldots,x_t)| &\leq L\sum_{j=1}^{H}\|x_t - x_{t-j}\| \leq L\sum_{j=1}^{H}\sum_{l=1}^{j}\|x_{t-l+1} - x_{t-l}\| \\
&\leq L\sum_{j=1}^{H}\sum_{l=1}^{j}\eta_{t-l}\|\nabla\tilde{f}_{t-l}(x_{t-l})\| \leq LH^2\eta_{t-H}G,
\end{aligned}$$

Algorithm 4 OGD with Memory (OGD-M).

Input: Step size η, functions $\{f_t\}_{t=m}^{T}$
Initialize $x_0, \ldots, x_{H-1} \in \mathcal{K}$ arbitrarily.
for $t = H, \ldots, T$ **do**
    Play x_t, suffer loss $f_t(x_{t-H}, \ldots, x_t)$
    Set $x_{t+1} = \Pi_{\mathcal{K}} \left(x_t - \eta \nabla \tilde{f}_t(x) \right)$
end for

and so we have that

$$\left| \sum_{t=H}^{T} f_t(x_{t-H}, \ldots, x_t) - \sum_{t=H}^{T} f_t(x_t, \ldots, x_t) \right| \leq \frac{LH^2G}{\lambda}(1 + \log(T)).$$

It follows that

$$\sum_{t=H}^{T} f_t(x_{t-H}, \ldots, x_t) - \min_{x \in \mathcal{K}} \sum_{t=H}^{T} f_t(x, \ldots, x) \leq \frac{G^2 + LH^2G}{\lambda}(1 + \log(T)).$$

□

3.9 Policy regret for online natural gradient descent

Recall the setting involving structured loss functions of the form $f_t(M_{0:H+1}) = \mathbb{E}[c_t(z)]$, where $z = \sum_{i=0}^{H+1} J_i [M_i]_{\text{vec}}$. J_i is a random matrix, and c_t's are adversarially chosen strongly convex loss functions. In a similar vein, define $f_t(M)$ to be the specialization of f_t when input the same argument, i.e. M, $H+1$ times. Define $J = \sum_{i=0}^{H+1} J_i$. The following lemma provides upper bounds on the regret bound as well as the norm of the movement of iterate at every round for the Online Natural Gradient Update (Algorithm 3).

Lemma 3.9.1. *For α-strongly convex c_t, if the iterates M_t are chosen as per the update rule:*

$$[M_{t+1}]_{vec} = \Pi_{\mathcal{M}} \left([M_t]_{vec} - \eta_t (\mathbb{E}[J^T J])^{-1} \nabla_{[M_t]_{vec}} f_t(M_t) \right)$$

with a decreasing step size of $n_t = \frac{1}{\alpha t}$, it holds that

$$\sum_{t=1}^{T} f_t(M_t) - \min_{M^* \in \mathcal{M}} \sum_{t=1}^{T} f_t(M^*) \leq (2\alpha)^{-1} \max_{M \in \mathcal{M}} \|\nabla_{M_{vec}} f_t(M)\|^2_{(\mathbb{E}[J^T J])^{-1}} \log T.$$

Moreover, the norm of the movement of consecutive iterates is bounded for all t as

$$\|[M_{t+1}]_{vec} - [M_t]_{vec}\| \leq (\alpha t)^{-1} \max_{M \in \mathcal{M}} \|(\mathbb{E}[J^T J])^{-1} \nabla_{M_{vec}} f_t(M)\|.$$

Proof. Let $M^* = \arg\min_{M \in \mathcal{M}} \sum_{t=1}^{T} f_t(M)$, $z_t = JM_{vec,t}$ and $z^* = JM^*_{\text{vec}}$. Now, we have, as consequence of strong convexity of c_t, that

$$\sum_{t=1}^{T} f_t(M_t) - \sum_{t=1}^{T} f_t(M^*) \leq \mathbb{E}\left[\langle \nabla_z c_t(z_t), z_t - z^* \rangle - \frac{\alpha}{2}\|z_t - z^*\|^2\right].$$

With $P = \mathbb{E}[J^T J]$, the choice of the update rule ensures that

$$\begin{aligned}
&\|[M_{t+1}]_{\text{vec}} - M^*_{vec}\|_P^2 \\
=&\|[M_t]_{\text{vec}} - M^*_{\text{vec}}\|_P^2 - 2\eta_t \langle \nabla_{[M_t]_{\text{vec}}} f_t(M_t), [M_t]_{\text{vec}} - M^*_{\text{vec}} \rangle + \eta_t^2 \|\nabla_{[M_t]_{\text{vec}}} f_t(M_t)\|_{P^{-1}}.
\end{aligned}$$

Observe by the application of chain rule and linearity of expectation that

$$\begin{aligned}
\mathbb{E}[\langle \nabla_z c_t(z_t), z_t - z^* \rangle] &= \mathbb{E}[\langle \nabla_z c_t(z_t), J([M_t]_{\text{vec}} - M^*_{\text{vec}}) \rangle] \\
&= \langle \nabla_{[M_t]_{\text{vec}}} f_t(M_t), [M_t]_{\text{vec}} - M^*_{\text{vec}} \rangle, \\
\mathbb{E}[\|z_t - z^*\|^2] &= \|[M_t]_{\text{vec}} - M^*_{\text{vec}}\|_P^2.
\end{aligned}$$

Combining these (in)equalities, we have

$$\begin{aligned}
&\sum_{t=1}^{T} f_t(M_t) - \sum_{t=1}^{T} f_t(M^*) \\
\leq &\sum_{t=1}^{T} \left(\frac{\|[M_t]_{\text{vec}} - M^*_{vec}\|_P^2 - \|[M_{t+1}]_{\text{vec}} - M^*_{vec}\|_P^2}{2\eta_t} + \frac{\eta_t}{2} \|\nabla_{[M_t]_{\text{vec}}} f_t(M_t)\|_{P^{-1}}^2 \right) - \frac{\alpha}{2} \|[M_t]_{vec} - M^*_{vec}\|_P^2 \\
\leq &(2\alpha)^{-1} \max_{M \in \mathcal{M}} \|\nabla_{M_{\text{vec}}} f_t(M)\|_{P^{-1}}^2 \log T
\end{aligned}$$

□

Proof of Theorem 3.5.4. We know by equation 3.3.1 that, for any $t \geq H$,

$$\begin{aligned}
|f_t(M_{t-H:t}) - f_t(M)| &\leq L \sum_{j=1}^{H} \|[M_t]_{\text{vec}} - [M_{t-j}]_{\text{vec}}\| \leq L \sum_{j=1}^{H} \sum_{l=1}^{j} \|[M_{t-l+1}]_{\text{vec}} - [M_{t-l}]_{\text{vec}}\| \\
&\leq L \sum_{j=1}^{H} \sum_{l=1}^{j} \eta_{t-l} \max_{M \in \mathcal{M}} \|(\mathbb{E}[J^T J])^{-1} \nabla_{M_{\text{vec}}} f_t(M)\| \\
&\leq LH^2 \eta_{t-H} \max_{M \in \mathcal{M}} \|(\mathbb{E}[J^T J])^{-1} \nabla_{M_{\text{vec}}} f_t(M)\|,
\end{aligned}$$

and so we have that

$$\left| \sum_{t=H}^{T} f_t(M_{t-H:t}) - \sum_{t=H}^{T} f_t(M_t) \right| \leq \frac{LH^2 G_f}{\alpha} (1 + \log(T)).$$

The result follows by invoking Lemma 3.9.1. □

3.10 Spectral Lower Bound on $\mathbb{G}$

Proof of Lemma 3.6.3. The following definitions help us express the matrix $\mathbb{G}$ in a more convenient form. For any number $\psi \in \mathbb{C}$, such that $|\psi| < 1$ and any h define,

$$S_\psi(h) = \sum_{i=1}^{h} |\psi|^{2(i-1)} = \frac{1 - |\psi|^{2h}}{1 - |\psi|^2}.$$

With the above definition it can be seen that the entries $\mathbb{G}(\psi)$ can be expressed in the following manner,

$$[\mathbb{G}(\psi)]_{ij} = S_\psi(H - |i-j|) \cdot \psi^{i-j} \qquad \text{if } j \geq i$$

$$[\mathbb{G}(\psi)]_{ij} = (\psi^\dagger)^{j-i} \cdot S_\psi(H - |i-j|) \qquad \text{if } i \geq j$$

Schematically the matrix $\mathbb{G}(\psi)$ looks like

$$\begin{bmatrix}
S_\psi(H) & S_\psi(H-1)\psi & S_\psi(H-2)\psi^2 & . & . & S(2)\psi^{H-2} & S(1)\psi^{H-1} \\
\psi^\dagger S_\psi(H-1) & S_\psi(H) & S_\psi(H-1)\psi & . & . & S(3)\psi^{H-3} & S(2)\psi^{H-2} \\
(\psi^\dagger)^2 S_\psi(H-2) & \psi^\dagger S_\psi(H-1) & S_\psi(H) & . & . & S(4)\psi^{H-4} & S(3)\psi^{H-3} \\
. & . & . & . & . & . & . \\
. & . & . & . & . & . & . \\
(\psi^\dagger)^{H-1} S_\psi(1) & (\psi^\dagger)^{H-2} S_\psi(2) & (\psi^\dagger)^{H-3} S_\psi(3) & . & . & \psi^\dagger S_\psi(H-1) & S_\psi(H)
\end{bmatrix}.$$

We analytically compute the inverse of the matrix $\mathbb{G}(\psi)$ below and bound its spectral norm.

Claim 3.10.1. *The inverse of $\mathbb{G}(\psi)$ has the following form.*

$$[\mathbb{G}(\psi)]^{-1} = \begin{bmatrix} \alpha & b & 0 & . & . & 0 & 0 & \beta^\dagger \\ b^\dagger & a & b & . & . & 0 & 0 & 0 \\ 0 & b^\dagger & a & . & . & 0 & 0 & . \\ . & 0 & b^\dagger & . & . & b & 0 & . \\ . & 0 & 0 & . & . & a & b & 0 \\ 0 & 0 & 0 & . & . & b^\dagger & a & b \\ \beta & 0 & 0 & . & . & 0 & b^\dagger & \alpha \end{bmatrix},$$

where the relevant quantities above are given by the following formula

$$b = \frac{-\psi}{1+|\psi|^{2H}} \qquad a = -b(\psi^\dagger + \psi^{-1}) = \frac{1+|\psi|^2}{1+|\psi|^{2H}}$$

$$\beta = \frac{(1-|\psi|^2)}{(1-(|\psi|^2)^{H+1})}\frac{(\psi^\dagger)^H\psi}{(1+|\psi|^{2H})} \qquad \alpha = \frac{1-(|\psi|^2)^{H+2}}{(1-(|\psi|^2)^{H+1})(1+(|\psi|^{2H}))}.$$

Since $|\psi| < 1$, it is easy to see that $|\alpha|, |a| \leq 2$ and $|\beta|, |b| \leq 1$. This immediately implies that $\|(\mathbb{G}(\psi))^{-1}\| \leq 4$ and therefore the lemma follows.

To prove the remnant claim, the following may be verified, implying $\mathbb{G}(\psi)[\mathbb{G}(\psi)]^{-1} = I$.

Case A: Let's first consider the diagonal entries and in particular $i = j \in [1, H-2]$. We have that

$$\begin{aligned} &\left[\mathbb{G}(\psi)[\mathbb{G}(\psi)]^{-1}\right]_{i,i} = b \cdot \psi^\dagger S_\psi(H-1) + b^\dagger \cdot \psi S_\psi(H-1) + aS_\psi(H) \\ =&\frac{-2|\psi|^2 S_\psi(H-1) + (1+|\psi|^2)S_\psi(H)}{1+|\psi|^{2H}} = 1 \end{aligned}$$

Case B: Lets consider the diagonal entry $(0,0)$. (The (H,H) entry is the complement and hence

equal to 1).

$$\begin{aligned}
&\left[\mathbb{G}(\psi)[\mathbb{G}(\psi)]^{-1}\right]_{0,0}\\
=&\alpha\cdot S_\psi(H)+b^\dagger\psi S_\psi(H-1)+\beta^\dagger(\psi^\dagger)^{H-1}S_\psi(1)\\
=&\frac{(1-(|\psi|^2)^{H+2})S_\psi(H)-(1-(|\psi|^2)^{H+1})|\psi|^2S_\psi(H-1)+(1-|\psi|^2)(|\psi|^{2H})}{(1-(|\psi|^2)^{H+1})(1+(|\psi|^{2H}))}\\
=&1
\end{aligned}$$

Case C: Now lets consider non diagonal entries, in particular for $j\in[1,H-2]$ and $i\in[0,H-1]$ and $i>j$. (The case with the same conditions and $j>i$ follows by replacing ψ with $\psi^\dagger$ in the computation below)

$$\begin{aligned}
\left[\mathbb{G}(\psi)[\mathbb{G}(\psi)]^{-1}\right]_{i,j}&=(\psi^\dagger)^{i-j-1}\left(b(\psi^\dagger)^2S_{H-i+j-1}+b^\dagger S_{H-i+j+1}+a(\psi^\dagger)S_{H-i+j}\right)\\
&=(\psi^\dagger)^{i-j}\left(-|\psi|^2S_{H-i+j-1}-S_{H-i+j+1}+(|\psi|^2+1)S_{H-i+j}\right)\\
&=0
\end{aligned}$$

Case D: Lastly lets consider the first column, i.e. $j=0$ and $i>0$. (The case of the last column follows as it is the complement and hence equal to 0.)

$$\begin{aligned}
&\left[\mathbb{G}(\psi)[\mathbb{G}(\psi)]^{-1}\right]_{i,j}\\
=&\alpha\cdot(\psi^\dagger)^iS_\psi(H-i)+b\cdot(\psi^\dagger)^{i-1}S_\psi(H-i+1)+\beta\psi^{H-i-1}S_\psi(i+1)\\
=&0.
\end{aligned}$$

□

3.11 Proof of strong convexity for arbitrary systems

Proof of Lemma 3.6.1. Building on Section 3.6, we prove Lemma 3.6.1 for multi-dimensional systems. Using the fact that $E[w_iw_j^\top]=0$ for different i,j and $\mathbb{E}[w_iw_i^\top]=\Sigma$, it can be observed that for any k_1,k_2 and any $d_u\times d_u$ matrix P, we have that

$$\mathbb{E}[J_{v\tilde{k}_1}^\top PJ_{v\tilde{k}_2}]=\mathcal{T}_{k_1-k_2}\otimes P\otimes\Sigma \tag{3.11.1}$$

where $\mathcal{T}_m$ is defined as an $H \times H$ matrix with $[\mathcal{T}_m]_{ij} = 1$ if and only if $i - j = m$ and 0 otherwise. This in particular immediately gives us that for any matrix P,

$$\begin{aligned}\mathbb{E}[J_y^\top P J_y] &= \left(\sum_{k_1=1}^{H}\sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \otimes \left(\left(B^\top (A-B\mathbb{K})^\top\right)^{k_1-1} P (A-B\mathbb{K})^{k_2-1} B\right)\right) \otimes \Sigma \\ &= \left((I_H \otimes B^\top) \underbrace{\left(\sum_{k_1=1}^{H}\sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \otimes \left(\left((A-B\mathbb{K})^\top\right)^{k_1-1} P (A-B\mathbb{K})^{k_2-1}\right)\right)}_{\triangleq \mathbb{G}_P} (I_H \otimes B) \right) \otimes \Sigma \end{aligned} \tag{3.11.2}$$

Furthermore consider the following calculation

$$\mathbb{E}[J_{\tilde{v}_0}^\top \mathbb{K} J_y] = \left(\sum_{k=1}^{H} \mathcal{T}_{-k} \otimes \mathbb{K}(A-B\mathbb{K})^{k-1} B\right) \otimes \Sigma \tag{3.11.3}$$

$$= \left((I_H \otimes \mathbb{K}) \underbrace{\left(\sum_{k=1}^{H} \mathcal{T}_{-k} \otimes (A-B\mathbb{K})^{k-1}\right)}_{\triangleq \mathbb{Y}} (I_H \otimes B) \right) \otimes \Sigma \tag{3.11.4}$$

As before, we state the following bounds on the spectral properties of the matrices $\mathbb{G}$ and $\mathbb{Y}$ defined above.

Lemma 3.11.1.

$$\|\mathbb{Y}\| \leq \|\sum_{k=1}^{H} \mathcal{T}_{-k}(A-B\mathbb{K})^{k-1}\| \leq \gamma^{-1}\kappa^2 \tag{3.11.5}$$

Lemma 3.11.2. $\mathbb{G}_I$ *(where I represents the Identity matrix) is a symmetric positive definite matrix with*

$$\mathbb{G}_I \succeq \frac{1}{4\kappa^4} \cdot I_{Hd_x}$$

Consider the following calculations which follows by definitions.

$$\begin{aligned}\mathbb{E}[J_v^\top J_v] &= \mathbb{E}[J_y^\top \mathbb{K}^\top \mathbb{K} J_y] - \mathbb{E}[J_y^\top \mathbb{K}^\top J_{\tilde{v}_0}] - \mathbb{E}[J_{\tilde{v}_0}^\top \mathbb{K} J_y] + \mathbb{E}[J_{\tilde{v}_0}^\top J_{\tilde{v}_0}] \\ &= \underbrace{\left((I_H \otimes B^\top)\mathbb{G}_{\mathbb{K}^\top\mathbb{K}}(I_H \otimes B) - \mathbb{Y}(I_H \otimes B) - (I_H \otimes B^\top)\mathbb{Y}^\top + I_{Hd_u}\right)}_{\triangleq \mathbb{F}} \otimes \Sigma\end{aligned}$$

Since we know that $\Sigma \succeq 0$ we immediately get that $\mathbb{F} \succeq 0$. Using the above calculations it is enough to show that the following matrix has lower bounded eigenvalues, i.e. for every vector m of appropriate

dimensions, we have that

$$m^\top \left(\mathbb{F} + (I_H \otimes B^\top)\mathbb{G}_I(I_H \otimes B)\right) m \geq \frac{\gamma^2 \|m\|^2}{36\kappa^{10}}$$

To prove the above we will consider two cases. The first case is when $\|(I_H \otimes B)m\| \geq \frac{\gamma\|m\|}{3\kappa^3}$. In this case note that

$$m^\top \left(\mathbb{F} + (I_H \otimes B^\top)\mathbb{G}_I(I_H \otimes B)\right) m \geq m^\top \left((I_H \otimes B^\top)\mathbb{G}_I(I_H \otimes B)\right) m \geq \frac{\frac{1}{4\kappa^4}\gamma^2\|m\|^2}{9\kappa^6}$$

In the second case (when $\|(I_H \otimes B)m\| \leq \frac{\gamma\|m\|}{3\kappa^3}$), we have that

$$\begin{aligned}
&m^\top \left(\mathbb{F} + (I_H \otimes B^\top)\mathbb{G}_I(I_H \otimes B)\right) m \\
&\geq m^\top \left(I_{Hd_u} - (I_H \otimes \mathbb{K})\mathbb{Y}(I_H \otimes B) - (I_H \otimes B^\top)\mathbb{Y}^\top(I_H \otimes \mathbb{K}^\top)\right) m \\
&\geq (1/3)\|m\|^2 \geq \frac{\gamma^2\|m\|^2}{36\kappa^{10}}.
\end{aligned}$$

□

We now finish the proof with the proof of Lemmas 3.11.1 and 3.11.2.

Proof of Lemma 3.11.1. Since $\mathbb{K}$ is (κ, γ)-diagonal strongly stable, we can diagonalize the matrix $A - B\mathbb{K}$ as $A - B\mathbb{K} = QLQ^{-1}$ with $\|Q\|, \|Q\|^{-1} \leq \kappa$. Therefore,

$$\mathbb{Y} = \left(\sum_{k=1}^{H} \mathcal{T}_{-k} \otimes QL^{k-1}Q^{-1}\right) = (I_H \otimes Q)\left(\sum_{k=1}^{H} \mathcal{T}_{-k} \otimes L^{k-1}\right)(I_H \otimes Q^{-1}).$$

Now consider the matrix P for any complex number ϕ with $|\phi| < 1$.

$$P = \sum_{k=1}^{H} \mathcal{T}_{-k}\phi^{k-1}$$

We wish to bound $\|P\|$. To this end consider $PP^\top$ and consider the ℓ_1 norm of any row. It can easily be seen that the ℓ_1 norm of any row of $PP^\top$ is bounded by $\frac{1}{1-|\phi|} \cdot \frac{1}{1-|\phi|^2}$, and therefore

$$\|P\| = \sqrt{\|PP^\top\|} \leq \sqrt{\frac{1}{(1-|\phi|)(1-|\phi|^2)}}.$$

Using that L is diagonal with entries bounded in magnitude by $1 - \gamma$, we get that $\|\mathbb{Y}\| \leq \gamma^{-1}\kappa^2$. □

Proof of Lemma 3.11.2. We need to consider the following matrix

$$\mathbb{G}_I = \sum_{k_1=1}^{H} \sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \otimes \left(\left((A - B\mathbb{K})^\top \right)^{k_1-1} (A - B\mathbb{K})^{k_2-1} \right)$$

Since $\mathbb{K}$ is (κ, γ)-diagonal strongly stable, we can diagonalize the matrix $A - B\mathbb{K}$ as $A - B\mathbb{K} = QLQ^{-1}$ with $\|Q\|, \|Q\|^{-1} \leq \kappa$. Further since $A - B\mathbb{K}$ is a real valued matrix we have that $(A - B\mathbb{K})^\top = (Q^{-1})^\dagger L^\dagger Q^\dagger$. Therefore we have that

$$\mathbb{G}_I = \sum_{k_1=1}^{H} \sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \otimes \left((Q^{-1})^\dagger \left(L^\dagger \right)^{k_1-1} Q^\dagger Q L^{k_2-1} Q^{-1} \right)$$

Further consider the following matrix $\hat{\mathbb{G}}$.

$$\hat{\mathbb{G}} = \begin{bmatrix} 0 & 0 & . & . & I \\ 0 & . & . & . & L \\ . & . & . & . & L^2 \\ . & 0 & . & . & . \\ 0 & I & . & . & . \\ I & L & . & . & L^{H-1} \\ L & L^2 & . & . & 0 \\ L^2 & . & . & . & 0 \\ . & . & . & . & . \\ . & L^{H-1} & . & . & . \\ L^{H-1} & 0 & . & . & 0 \end{bmatrix}$$

It can be seen that,

$$\left((I_{2H-1} \otimes Q) \hat{\mathbb{G}} (I_{2H-1} \otimes Q^{-1}) \right)^\dagger \left((I_{2H-1} \otimes Q) \hat{\mathbb{G}} (I_{2H-1} \otimes Q^{-1}) \right) = \mathbb{G}_I. \tag{3.11.6}$$

Furthermore note that since $\|Q\|, \|Q^{-1}\| \leq \kappa$, therefore all singular values of Q lie in the range $[\kappa^{-1}, \kappa]$. Therefore it follows that

$$Q^\dagger Q \succeq \kappa^{-2} I \qquad (Q^{-1})^\dagger Q^{-1} \succeq \kappa^{-2} I \tag{3.11.7}$$

Using equations 3.11.6 and 3.11.7 it follows that

$$\mathbb{G}_I \succeq \kappa^{-4} \cdot \left(\hat{\mathbb{G}}\right)^{\dagger} \left(\hat{\mathbb{G}}\right) \tag{3.11.8}$$

Therefore we only need to show that $\left(\hat{\mathbb{G}}\right)^{\dagger} \left(\hat{\mathbb{G}}\right)$ has a lower bounded eigenvalue. To that end notice that since L is a diagonal matrix with diagonal values whose magnitude is upper bounded by 1. Therefore, it sufficient to consider the case when L is a scalar complex number with magnitude upper bounded by 1. To this end we can consider the following simplification of $\mathbb{G}_I$ defined for a complex number ψ with $|\psi| < 1$ as defined earlier.

$$\mathbb{G}(\psi) = \sum_{k_1=1}^{H} \sum_{k_2=1}^{H} \mathcal{T}_{k_1-k_2} \left(\psi^{\dagger}\right)^{k_1-1} \psi^{k_2-1}$$

Invoking Lemma 3.6.3 we immediately get that

$$\mathbb{G}_I \succeq \kappa^{-4} \cdot \left(\hat{\mathbb{G}}\right)^{\dagger} \left(\hat{\mathbb{G}}\right) \succeq \frac{1}{4\kappa^4} \cdot I_{Hd_x}.$$

□

Chapter 4

Extension to Unknown Systems

This chapter extends the nonstochastic control methodology to the case of unknown system dynamics. While in the previous chapters knowledge of the system matrices (A, B) was assumed, here the objective is to control an unknown linear dynamical system in the presence of (nonstochastic) adversarial perturbations and adversarial convex loss functions. The main contribution in this chapter is the first efficient algorithm that guarantees a sublinear regret bound, scaling as $O(T^{2/3})$ for the unknown system case.

4.1 Introduction

In the prevous chapters, we considered the task of controlling a linear dynamical system, where the state evolved at every timestep as a linear transformation of the previous state, control input and a nonstochastic disturbance. It was assumed that the learner or the algorithm has full knowledge of this linear transformation.

$$x_{t+1} = Ax_t + Bu_t + w_t$$

As implied by the sublinear regret guarantee, the implication of the existence of such control algorithms is that the average excess cost of not knowing the distrubance and cost sequences in advance is vanishing. The present chapter asks if an analogous statement holds while the state-state and state-action (A, B respectively) system matrices that specify the above-mentioned linear transform are not known in advance.

The main result in this chapter is an efficient algorithm for this setting which attains the following guarantee:

Theorem 4.1.1 (Informal Statement)**.** *For an unknown linear dynamical system where the perturbations w_t (and convex costs c_t) are bounded and chosen by an adversary, there exists an efficient algorithm that generates an adaptive sequence of controls $\{u_t\}$ for which*

$$\textit{Regret} = O(\text{poly}(\texttt{natural-parameters})T^{2/3}).$$

Our starting point is the result presented in chapter 1, first published in [ABH+19], where a novel class of policies choosing actions as a linear combination of past perturbations, $u_t = \sum_{i=1}^{k} M_i w_{t-i}$ was proposed. It was demonstrated that learning the coefficients M_i, via online convex optimization, allows the resultant controller to compete with the class of all linear state-feedback policies. This latter class is important, since it is known to be optimal for the standard setting of normal i.i.d noise and quadratic loss functions, also known as the Linear Quadratic Regulator (LQR), and associated robust control settings (see [BB08] for examples).

The caveat in [ABH+19] is that the system matrices (A, B) needed to be known. In the case of a known system, the disturbances can be simply computed via observations of the state, ie. $w_t = x_{t+1} - Ax_t - Bu_t$. This chapter takes the approach of identifying the system, or the matrices A, B, from observations, and thereafter appealing to the inherent robustness of the previously proposed control approaches. This is non-trivial since the noise is assumed to be adversarial, and was posed as a question in [Tu19].

In this chapter, we show how to overcome this difficulty and obtain sublinear regret for controlling an unknown system in the presence of adversarial noise and adversarial loss functions. The regret notion we adopt is policy regret against linear policies, exactly as in previous chapters. An important component that we use is *adversarial sys-id*: an efficient method for uncovering the underlying system even in the presence of adversarial perturbations. This method is not based on naive least squares method of regressing (x_t, u_t) on x_{t+1}. In particular, without independent, zero-mean w_t's, the latter approach can produce inconsistent estimates of the system matrices.

4.1.1 Related work

Robust Control: The classical control literature deals with adversarial perturbations in the dynamics in a framework known as H_∞ control, see e.g. [Ste94a, ZDG+96b]. In this setting, the

controller solves for the best linear controller assuming worst case noise to come, i.e.,

$$\min_{K_1} \max_{w_1} \min_{K_2} ... \min_{K_T} \max_{w_T} \sum_t c_t(x_t, u_t).$$

This approach is overly pessimistic as it optimizes over the worst-case noise. In contrast, the metric in nonstochastic control (NSC) is regret, which adapts to the per-instance perturbations.

Learning to control stochastic LDS: There has been a resurgence of literature on control of linear dynamical systems in the recent machine learning venues. The case of known systems was extensively studied in the control literature, see the survey [Ste94a]. Sample complexity and regret bounds for control (under Gaussian noise) were obtained in [AYS11, DMM$^+$18, AYLS19, MTR19, CKM19]. The works of [AYBK14], [CHK$^+$18] and [AHS19] allow for control in LDS with adversarial loss functions. Provable control in the Gaussian noise setting via the policy gradient method was studied in [FGKM18]. These works operate in the absence of perturbations or assume that the same are i.i.d., as opposed to our adversarial.

Control with adversarial perturbations: The most relevant reformulation of the control problem that enables our result is the recent work of [ABH$^+$19], who use online learning techniques and convex relaxation to obtain provable bounds for controlling LDS with adversarial perturbations. However, the result and the algorithm make extensive use of the availability of the system matrices. [Tu19] ask if the latter result can be extended to unknown systems, a question that we answer in the affirmative.

System identification. For the stochastic setting, several works [FTM18, SMT$^+$18, SR19] propose to use the least-squares procedure for parameter identification. In the adversarial setting, least-squares can lead to inconsistent estimates. For the partially observed stochastic setting, [OO19, SRD19, SMT$^+$18] give results guaranteeing parameter recovery using Gaussian inputs. Of these, the results in [SBR19] also apply to the adversarial setting. We offer a simpler analysis for parameter identification, and develop rigorous perturbation bounds for the control algorithm necessary to make guarantees on the quality of the control solution. Other relevant work from the machine learning literature includes spectral filtering techniques for learning and open-loop control of partially observable systems [HSZ17b, AHL$^+$18, HLS$^+$18].

4.2 Problem definition

We consider the setting of linear dynamical systems with time-invariant dynamics, i.e.

$$x_{t+1} = Ax_t + Bu_t + w_t,$$

where $x_t \in \mathbb{R}^m$ and $u_t \in \mathbb{R}^n$. The perturbation sequence w_t may be adversarially chosen at the beginning of the interaction, and is unknown to the learner. Likewise, the system is augmented with time-varying convex cost functions $c_t(x, u)$. The total cost associated with a sequence of (random) controls, derived through an algorithm $\mathcal{A}$, is

$$J(\mathcal{A}) = \sum_{t=1}^{T} c_t(x_t, u_t).$$

With some abuse of notation, we will denote by $J(K)$ the cost associated with the execution of controls as a linear controller K would suggest, ie. $u_t = -Kx_t$. The following conditions are assumed on the cost and the perturbations [1].

Assumption 4.2.1. *The perturbation sequence is bounded, ie. $\|w_t\| \leq W$, and chosen at the start of the interaction, implying that this sequence w_t does not depend on the choice of u_t.*

Assumption 4.2.2. *As long as $\|x_t\|, \|u_t\| \leq D$, the convex costs admit $\|\nabla_{(x,u)} c_t(x, u)\| \leq GD$.*

The fundamental Linear Quadratic Regulator problem is a specialization of the above to the case when the perturbations are i.i.d. Gaussian and the cost functions are positive quadratics, ie.

$$c_t(x, u) = x^\top Qx + u^\top Ru.$$

Objective We consider the setting where the learner has no knowledge of A, B and the perturbation sequence w_t. In this case, any inference of these quantities may only take place indirectly through the observation of the state x_t. Furthermore, the learner is made aware of the cost function c_t only once the choice of u_t has been made.

Under such constraints, the objective of the algorithm is to choose an (adaptive) sequence of controls that ensure that the cost suffered in this manner is comparable to that of the best choice of a linear controller with complete knowledge of system dynamics A, B and the foreknowledge of the

[1] Without loss of generality we shall assume that $G, D, W, \kappa \geq 1$ holds, since these are upper bounds.

cost and perturbation sequences $\{c_t, w_t\}$. Formally, we measure regret as

$$\text{Regret} = J(\mathcal{A}) - \min_{K \in \mathcal{K}} J(K).$$

$\mathcal{K}$ is the set of (κ, γ)-strongly stable linear controllers defined below. The notion of strong stability, introduced in [CHK$^+$18], offers a quantification of the classical notion of a stable controller in manner that permits a discussion on non-asymptotic regret bounds.

Definition 4.2.3 (Strong Stability)**.** *A linear controller K is (κ, γ)-strongly stable for a linear dynamical system specified via (A, B) if there exists a decomposition of $A - BK = QLQ^{-1}$ with $\|L\| \leq 1 - \gamma$, and $\|A\|, \|B\|, \|K\|, \|Q\|, \|Q^{-1}\| \leq \kappa$.*

We also assume the learner has access to a fixed stabilizing controller $\mathbb{K}$. When operating under unknown transition matrices, the knowledge of a stabilizing controller permits the learner to prevent an inflation of the size of the state beyond reasonable bounds.

Assumption 4.2.4. *The learner knows a linear controller $\mathbb{K}$ that is (κ, γ)-strongly stable for the true, but unknown, transition matrices (A, B) defining the dynamical system.*

The non-triviality of the regret guarantee rests on the benchmark set not being empty. As noted in [CHK$^+$18], a sufficient condition to ensure the existence of a strongly stable controller is the controllability of the linear system (A, B). Informally, controllability for a linear system is characterized by the ability to drive the system to any desired state through appropriate control inputs in the presence of deterministic dynamics, ie. $x_{t+1} = Ax_t + Bu_t$.

Definition 4.2.5 (Strong Controllability)**.** *For a linear dynamical system (A, B), define, for $k \geq 1$, a matrix $C_k \in \mathbb{R}^{n \times km}$ as*

$$C_k = [B, AB, A^2B \dots A^{k-1}B].$$

A linear dynamical system (A, B) is controllable with controllability index k if C_k has full row-rank. In addition, such a system is also (k, κ)-strongly controllable if $\|(C_k C_k^\top)^{-1}\| \leq \kappa$.

As with stability, a quantitative analog of controllability first suggested in [CHK$^+$18] is presented above. It is useful to note that, as a consequence of the Cayley-Hamiltion theorem, for a controllable system the controllability index is always at most the dimension of the state space. We adopt the assumption that the system $(A - B\mathbb{K}, B)$ is (k, κ) strongly controllable.

Assumption 4.2.6. *The linear dynamical system $(A - B\mathbb{K}, B)$ is (k, κ)-strongly controllable.*

4.3 Preliminaries

This section sets up the concepts that aid the algorithmic description and the analysis.

4.3.1 Parameterization of the controller

The total cost objective of a linear controller is non-convex in the canonical parameterization [FGKM18], ie. $J(K)$ is not convex in K. To remedy this, we use an alternative perturbation-based parameterization for controller, recently proposed in [ABH+19], where the advised control is linear in the past perturbations (as opposed to the state). This permits that the offline search for an optimal controller may be posed as a convex program.

Definition 4.3.1. *A perturbation-based policy $M = (M^{[0]}, \dots M^{[H-1]})$ chooses control u_t at state x_t,*

$$u_t = -\mathbb{K}x_t + \sum_{i=1}^{H} M^{[i-1]} w_{t-i}.$$

4.3.2 State evolution

Under the execution of a stationary policy M, the state may be expressed as a linear transformation Ψ (defined below) of the perturbations, where the transformation Ψ's is linear in the matrices M's.

$$x_{t+1} = (A - B\mathbb{K})^{H+1} x_{t-H} + \sum_{i=0}^{2H} \Psi_i(M|A,B) w_{t-i}$$

Definition 4.3.2. *For a matrix pair (A, B), define the state-perturbation transfer matrix:*

$$\Psi_i(M|A,B) = (A - B\mathbb{K})^i \mathbf{1}_{i \le H} + \sum_{j=0}^{H} (A - B\mathbb{K})^j B M^{[i-j-1]} \mathbf{1}_{i-j \in [1,H]}.$$

Definition 4.3.3. *Define the surrogate state y_{t+1} and the surrogate action v_t as stated below. The surrogate cost f_t as chosen to be the specialization of the t-th cost function with the surrogate state-action pair as the argument.*

$$\begin{aligned}
y_{t+1}(M|A,B,\{w\}) &= \sum_{i=0}^{2H} \Psi_i(M|A,B) w_{t-i} \\
v_t(M|A,B,\{w\}) &= -\mathbb{K} y_t(M|A,B,\{w\}) + \sum_{i=1}^{H} M^{[i-1]} w_{t-i} \\
f_t(M|A,B,\{w\}) &= c_t(y_t(M|A,B,\{w\}), v_t(M|A,B,\{w\}))
\end{aligned}$$

4.4 The algorithm

Our approach follows the explore-then-commit paradigm, identifying the underlying the deterministic-equivalent dynamics to within some accuracy using random inputs in the exploration phase. Such an approximate recovery of parameters permits an approximate recovery of the perturbations, thus facilitating the execution of the perturbation-based controller on the approximated perturbations.

Algorithm 5 Adversarial control via system identification.

Input: learning rate η, horizon H, number of iterations T, rounds of exploration T_0.
Phase 1: System Identification.
Call Algorithm 6 with a budget of T_0 rounds to obtain system estimates $\hat{A}, \hat{B}$.
Phase 2: Robust Control.
Define the constraint set $\mathcal{M}$ as $\mathcal{M} = \{M = (M^{[0]}, \dots M^{[H-1]}) : \|M^{[i-1]}\| \le \kappa^4(1-\gamma)^i\}$.
Initialize $\hat{w}_{T_0} = x_{T_0+1}$ and $\hat{w}_t = 0$ for $t < T_0$.
for $t = T_0 + 1, \dots, T$ **do**
  Choose the action

$$u_t = -\mathbb{K}x_t + \sum_{i=1}^{H} M_t^{[i-1]} \hat{w}_{t-i}.$$

  Observe the new state x_t, the cost function $c_t(x, u)$.
  Record an estimate $\hat{w}_t = x_{t+1} - \hat{A}x_t - \hat{B}u_t$.
  Update $M_{t+1} = \Pi_{\mathcal{M}}(M_t - \eta \nabla f_t(M_t | \hat{A}, \hat{B}, \{\hat{w}\}))$.
end for

Algorithm 6 System identification via random inputs.

Input: number of iterations T_0.
for $t = 0, \dots, T_0$ **do**
  Execute the control $u_t = -\mathbb{K}x_t + \eta_t$ with $\eta_t \sim_{i.i.d.} \{\pm 1\}^n$.
  Record the observed state x_t.
end for
Declare $N_j = \frac{1}{T_0 - k} \sum_{t=0}^{T_0-k-1} x_{t+j+1} \eta_t^\top$, for all $j \in [k]$.
Define $C_0 = (N_0, \dots N_{k-1})$, $C_1 = (N_1, \dots N_k)$, and return $\hat{A}, \hat{B}$ as

$$\hat{B} = N_0, \quad \hat{A}' = C_1 C_0^\top (C_0 C_0^\top)^{-1}, \quad \hat{A} = \hat{A}' + \hat{B}\mathbb{K}.$$

Theorem 4.4.1. *Under the assumptions 4.2.1, 4.2.2, 4.2.6, 4.2.4, when* $H = \Theta(\gamma^{-1} \log(\kappa^2 T))$, $\eta = \Theta(GW\sqrt{T})^{-1}$, $T_0 = \Theta(T^{2/3} \log \delta^{-1})$, *the regret incurred by Algorithm 5 for controlling an unknown linear dynamical system admits the upper bound* [2] *stated below with probability at least* $1 - \delta$.

$$\textit{Regret} = O(\text{poly}(\kappa, \gamma^{-1}, k, m, n, G, W) T^{2/3} \log \delta^{-1})$$

[2] If one desires the regret to scale as GW^2, as may be rightly demanded due to Assumption 4.2.2, it suffices to choose the exploration scheme in Algorithm 6 as $\{\pm W\}^n$, while introducing a multiplicative factor of W^{-2} in the computation N_j's. Such modifications ensure a natural scaling for all terms involving costs.

4.5 Regret analysis

To present the proof concisely, we set up a few articles of use. For a generic algorithm $\mathcal{A}$ operating on a generic linear dynamical system specified via a matrix pair (A, B) and perturbations $\{w\}$, let

1. $J(\mathcal{A}|A, B, \{w\})$ be the cost of executing $\mathcal{A}$, as incurred on the last $T - T_0$ time steps,
2. $x_t(\mathcal{A}|A, B, \{w\})$ be the state achieved at time step t, and
3. $u_t(\mathcal{A}|A, B, \{w\})$ be the control executed at time step t.

We also note that following result from [ABH+19] that applies to the case when the matrices (A, B) that govern the underlying dynamics are made known to the algorithm.

Theorem 4.5.1 (Known System; [ABH+19])**.** *Let $\mathbb{K}$ be a (κ, γ)-strong stable controller for a system (A, B), $\{w_t\}$ be a (possibly adaptive) perturbation sequence with $\|w_t\| \le W$, and c_t be costs satisfying Assumption 4.2.2. Then there exists an algorithm $\mathcal{A}$ (Algorithm 5.2 with a learning rate of $\eta = \Theta(GW\sqrt{T})^{-1}$ and $H = \Theta(\gamma^{-1}\log(\kappa^2 T))$), utilizing $\mathbb{K}, (A, B)$, that guarantees*

$$J(\mathcal{A}|A, B, \{w\}) - \min_{K\in\mathcal{K}} J(K|A, B, \{w\}) \le O(\mathrm{poly}(m, n, \kappa, \gamma^{-1})GW^2\sqrt{T}\log T).$$

Proof. of Theorem 4.4.1. Define $K = \mathrm{argmin}_{K\in\mathcal{K}} J(K)$. Let J_0 be the contribution to the *regret* associated with the first T_0 rounds of exploration. By Lemma 4.6.2, we have that $J_0 \le 16T_0Gn\kappa^8\gamma^{-2}W^2$.

Let $\mathcal{A}$ refer to the algorithm, from [ABH+19], executed in Phase 2. By Lemma 4.5.2,

$$\begin{aligned}\text{Regret} &\le J_0 + \sum_{t=T_0+1}^{T} c_t(x_t, u_t) - J(K|A, B, \{w\}),\\ &\le J_0 + (J(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) - J(K|\hat{A}, \hat{B}, \{\hat{w}\})) + (J(K|\hat{A}, \hat{B}, \{\hat{w}\}) - J(K|A, B, w)).\end{aligned}$$

Let $\|A - \hat{A}\|, \|B - \hat{B}\| \le \varepsilon_{A,B}$ and $\varepsilon_{A,B} \le 10^{-3}\kappa^{-10}\gamma^2$ in the arguments below. The middle term above can be upper bounded by the regret of algorithm $\mathcal{A}$ on the fictitious system $(\hat{A}, \hat{B})$ and the perturbation sequence $\{\hat{w}\}$. Before we can invoke Theorem 4.5.1, observe that

1. By Lemma 4.5.3, $\mathbb{K}$ is $(2\kappa, 0.5\gamma)$-strongly stable on $(\hat{A}, \hat{B})$, as long as $\varepsilon_{A,B} \le 0.25\kappa^{-3}\gamma$,
2. Lemma 4.5.6 ensures $\|\hat{w}_t\| \le 2\sqrt{n}\kappa^3\gamma^{-1}W$, as long as $\varepsilon_{A,B} \le 10^{-3}\kappa^{-10}\gamma^2$.

With the above observations in place, Theorem 4.5.1 guarantees

$$J(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) - J(K|\hat{A}, \hat{B}, \{\hat{w}\}) \le \mathrm{poly}(m, n, \kappa, \gamma^{-1})GW^2\sqrt{T}\log T.$$

The last expression in the preceding line can be bound by Lemma 4.5.4, since Lemma 4.5.6 also bounds $\|w_t - \hat{w}_t\| \le 20\sqrt{n}\kappa^{11}\gamma^{-3}W\varepsilon_{A,B}$ whenever $\varepsilon_{A,B} \le 10^{-3}\kappa^{-10}\gamma^2$ for $t \ge T_0 + 1$.

$$|J(K|\hat{A}, \hat{B}, \{\hat{w}\}) - J(K|A, B, w)| \le 32Gn\kappa^{11}\gamma^{-3}W^2 + 20^3TGn\kappa^{22}\gamma^{-7}W^2\varepsilon_{A,B}$$

Set $\varepsilon_{A,B} = \min(10^{-3}\kappa^{-10}\gamma^2, T^{-\frac{1}{3}})$ in Theorem 4.6.1 to conclude. □

The regret minimizing algorithm, Phase 2 of Algorithm 5, chooses M_t so as to optimize for the cost of the perturbation-based controller on a *fictitious* linear dynamical system $(\hat{A}, \hat{B})$ subject to the perturbation sequence $\{\hat{w}\}$. The following lemma shows that the definition of $\hat{w}_t$ ensures that state-control sequence visited by Algorithm 5 coincides with the sequence visited by the regret-minimizing algorithm on the *fictitious* system.

Lemma 4.5.2 (Simulation Lemma)**.** *Let $\mathcal{A}$ be the algorithm, from [ABH+19], executed in Phase 2, and (x_t, u_t) be the state-control iterates produced by Algorithm 5. Then for $t \ge T_0 + 1$,*

$$x_t = x_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}), \quad u_t = u_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}), \quad \textit{and} \quad \sum_{t=T_0+1}^{T} c_t(x_t, u_t) = J(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}).$$

Proof. This proof follows by induction on $x_t = x_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\})$. Note that at the start of $\mathcal{A}$, it is fed the initial state $x_{T_0+1}(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) = x_{T_0+1}$ by the choice of $\hat{w}_{T_0}$. Say that for some $t \ge T_0 + 1$, it happens that the inductive hypothesis is true. Consequently,

$$u_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) = -\mathbb{K}x_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) + \sum_{i=1}^{H} M_t^{[i-1]}\hat{w}_{t-i} = -\mathbb{K}x_t + \sum_{i=1}^{H} M_t^{[i-1]}\hat{w}_{t-i} = u_t.$$

$$x_{t+1}(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) = \hat{A}x_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) + \hat{B}u_t(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) + \hat{w}_t = \hat{A}x_t + \hat{B}u_t + \hat{w}_t = x_{t+1}$$

This, in turn, implies by choice of $\hat{w}_t$ that the next states produced at the next time steps match. □

The lemma stated below guarantees that the strong stability of $\mathbb{K}$ is approximately preserved under small deviations of the system matrices.

Lemma 4.5.3 (Preservation of Stability)**.** *If $\mathbb{K}$ is (κ, γ)-strongly stable for a linear system (A, B), ie. $A - BK = QLQ^{-1}$, then $\mathbb{K}$ is $(\kappa + \varepsilon_{A,B}, \gamma - 2\kappa^3\varepsilon_{A,B})$-strongly stable for $(\hat{A}, \hat{B})$, ie.*

$$\hat{A} - \hat{B}\mathbb{K} = Q\hat{L}Q^{-1}, \quad \|\hat{A}\|, \|\hat{B}\| \le \kappa + \varepsilon_{A,B}, \quad \|\hat{L}\| \le 1 - \gamma + 2\kappa^3\varepsilon_{A,B},$$

as long as $\|A - \hat{A}\|, \|B - \hat{B}\| \le \varepsilon_{A,B}$. Furthermore, in such case, the transforming matrices Q that

certify strong stability in both these cases coincide, and it holds $\|\hat{L} - L\| \leq 2\kappa^3 \varepsilon_{A,B}$.

Proof. Let $A - B\mathbb{K} = QLQ^{-1}$ with $\|Q\|, \|Q^{-1}\|, \|\mathbb{K}\|, \|A\|, \|B\| \leq \kappa$, $\|L\| \leq 1 - \gamma$. Now

$$\begin{aligned}\hat{A} - \hat{B}\mathbb{K} &= QLQ^{-1} + (\hat{A} - A) - (\hat{B} - B)\mathbb{K} \\ &= Q(L + Q^{-1}((\hat{A} - A) - (\hat{B} - B)\mathbb{K})Q)Q^{-1}\end{aligned}$$

It suffices to note that $\|Q^{-1}((\hat{A} - A) - (\hat{B} - B)\mathbb{K})Q\| \leq 2\kappa^3 \varepsilon_{A,B}$. □

The next lemma establishes that if the same linear state-feedback policy is executed on the actual and the *fictional* linear dynamical system, the difference between the costs incurred in the two scenarios varies proportionally with some measure of distance between the two systems.

Lemma 4.5.4 (Stability of Value Function)**.** *Let* $\|A - \hat{A}\|, \|B - \hat{B}\| \leq \varepsilon_{A,B} \leq 0.25\kappa^{-3}\gamma$*, and* K *be any* (κ, γ)*-strongly stable controller with respect to* (A, B)*. Then, for any perturbation sequence that satisfies* $\|w_t - \hat{w}_t\| \leq \varepsilon_w \leq W_0$ *except possibly for the first step where* $\|\hat{w}_0\| \leq {W_0}^3$*, it holds*

$$|J(K|\hat{A}, \hat{B}, \{\hat{w}\}) - J(K|A, B, w)| \leq 10^3 TG\kappa^8\gamma^{-3}W_0(\varepsilon_W + W_0\varepsilon_{A,B}) + 32G\kappa^5\gamma^{-2}W_0^2.$$

Proof. Under the action of a linear controller K, which is (κ, γ)-strongly stable for (A, B), it holds

$$x_{t+1}(K|A, B, \{w\}) = \sum_{t=0}^{T} (A - BK)^i w_{t-i}.$$

Consequently, $\|x_t(K|A, B, \{w\})\| \leq \kappa^2\gamma^{-1}W$, and, since K is $(2\kappa, 2^{-1}\gamma)$-strongly stable for $(\hat{A}, \hat{B})$ by Lemma 4.5.3, $\|x_t(K|\hat{A}, \hat{B}, \{\hat{w}\})\| \leq 16\kappa^2\gamma^{-1}W_0$. It follows

$$|J(\mathcal{A}|\hat{A}, \hat{B}, \{\hat{w}\}) - J(\mathcal{A}|A, B, w)| \leq 32G\kappa^3\gamma^{-1}W_0 \sum_{t=0}^{T} \|x_{t+1}(K|A, B, \{w\}) - x_{t+1}(K|\hat{A}, \hat{B}, \{\hat{w}\})\|$$

Finally, by using strong stability of the controller, we have

$$\begin{aligned}&\|x_{t+1}(K|A, B, \{w\}) - x_{t+1}(K|\hat{A}, \hat{B}, \{\hat{w}\})\| \leq \sum_{t=0}^{T} \left\|(A - BK)^i w_{t-i} - (\hat{A} - \hat{B}K)^i \hat{w}_{t-i}\right\| \\ \leq &\sum_{t=0}^{T} \left(\left\|(A - BK)^i w_{t-i} - (A - BK)^i \hat{w}_{t-i}\right\| + \left\|(A - BK)^i \hat{w}_{t-i} - (\hat{A} - \hat{B}K)^i \hat{w}_{t-i}\right\|\right) \\ \leq &\kappa^2(1 - \gamma)^t W_0 + \kappa^2\gamma^{-1}\varepsilon_w + 16W_0\kappa^5\gamma^{-2}\varepsilon_{A,B}\end{aligned}$$

[3]This handles the case for $\hat{w}_{T_0}$ in Algorithm 5. Assume that $W_0 \geq W$.

The last line follows from invocation of Lemma 4.5.5 on L, L' matrices that certify the strongly stability of K on (A, B) and $(\hat{A}, \hat{B})$ respectively. □

Lemma 4.5.5. *For any matrix pair $L, \Delta L$, such that $\|L\|, \|L + \Delta L\| \leq 1 - \gamma$, we have*

$$\sum_{t=0}^{\infty} \|(L + \Delta L)^t - L^t\| \leq \gamma^{-2} \|\Delta L\|$$

Proof. We make the inductive claim that $\|(L + \Delta L)^t - L^t\| \leq t(1 - \gamma)^{t-1} \|\Delta L\|$. The truth of this claim for $t = 0, 1$ is easily verifiable. Assuming the inductive claim for some t, observe

$$\|(L + \Delta L)^{t+1} - L^{t+1}\| \leq \|L((L + \Delta L)^t - L^t)\| + \|\Delta L(L + \Delta L)^t\| \leq (t + 1)(1 - \gamma)^t \|\Delta L\|.$$

Finally, observe that $\sum_{t=0}^{\infty} t(1 - \gamma)^{t-1} \leq \gamma^{-2}$. □

Lemma 4.5.6. *During Algorithm 5.2, it holds, as long as $\varepsilon_{A,B} \leq 10^{-3} \kappa^{-10} \gamma^2$, for any $t \geq T_0 + 1$*

$$\|x_t\| \leq 4\sqrt{n}\kappa^{10}\gamma^{-3}W, \quad \|w_t - \hat{w}_t\| \leq 20\sqrt{n}\kappa^{11}\gamma^{-3}W\varepsilon_{A,B}, \quad \textit{and} \quad \|\hat{w}_{t-1}\| \leq 2\sqrt{n}\kappa^3\gamma^{-1}W.$$

Proof. For a linear system evolving as as $x_{t+1} = Ax_t + Bu_t + w_t$, if the control is chosen as $u_t = -\mathbb{K}x_t + \tilde{u}_t$, the following is true, where $A' = A - B\mathbb{K}$.

$$x_{t+1} = \sum_{i=0}^{t} (A')^{t-i}(w_i + B\tilde{u}_i). \tag{4.5.1}$$

In Phase 2, $\tilde{u}_t$ is chosen as $\tilde{u}_t = \sum_{i=1}^{H} M_t^{[i-1]} \hat{w}_{t-i}$. We put forward the inductive hypothesis that for all $t \in [T_0 + 1, t_0]$, we have that $\|x_t\| \leq X := 4\sqrt{n}\kappa^{10}\gamma^{-3}W$, $\|\hat{w}_t - w_t\| \leq Y := 20\sqrt{n}\kappa^{11}\gamma^{-3}W\varepsilon_{A,B}$. If so, $\|\hat{w}_t\| \leq \max(W + Y, \|\hat{w}_{T_0}\|) = W + Y + \sqrt{n}\kappa^3\gamma^{-1}W$ for all $t \leq t_0$, by Lemma 4.6.2. The following display completes the induction.

$$\begin{aligned}
\|x_{t_0+1}\| &\leq \kappa^2\gamma^{-1}(W + \kappa^5\gamma^{-1}(W + Y + \sqrt{n}\kappa^3\gamma^{-1}W)) \leq X \\
\|\hat{w}_{t_0+1} - w_{t_0+1}\| &= \|(A - \hat{A})x_{t_0+1} + (B - \hat{B})u_{t_0+1}\| \\
&\leq \varepsilon_{A,B}(X + (\kappa X + \kappa^4\gamma^{-1}(W + Y + \sqrt{n}\kappa^3\gamma^{-1}W))) \leq Y
\end{aligned}$$

The base case be verified via computation. □

4.6 System identifcation via random inputs

This section details the guarantees afforded by the system identification procedure, which attempts to identify the deterministic-equivalent dynamics (A, B) by first identifying matrices of the form $(A')^i B$, where $A' = A - B\mathbb{K}$, and then recovering A by solving a linear system of equations.

Theorem 4.6.1 (System Recovery)**.** *Under assumptions 4.2.1, 4.2.6, 4.2.4, when Algorithm 6 is run for T_0 steps, the output pair $(\hat{A}, \hat{B})$ satisfies, with probability $1-\delta$, that $\|\hat{A} - A\|_F, \|\hat{B} - B\|_F \leq \varepsilon_{A,B}$, where*

$$T_0 = 10^3 kmn^2\kappa^{10}\gamma^{-2}W^2\varepsilon_{A,B}^{-2}\log(kmn\delta^{-1}).$$

Proof. Observe that A' is a unique solution of the system of equations (in X) presented below.

$$AC_k = [A'B, (A')^2B, \dots (A')^kB] = X[B, A'B, \dots (A')^{k-1}B] = XC_k$$

Now, if, for all j, $\|N_j - (A')^jB\| \leq \varepsilon$, it follows that $\|C_0 - C_k\|_F, \|C_1 - AC_k\|_F \leq \varepsilon\sqrt{k}$; in addition, $\|\hat{B} - B\| \leq \varepsilon$. By Lemma 4.6.4 on rows, we have

$$\|\hat{A}' - A'\|_F \leq \frac{2\varepsilon\sqrt{km}\kappa}{\sigma_{min}(C_k) - \varepsilon\sqrt{k}}.$$

So, setting $\varepsilon = \frac{\varepsilon_{A,B}}{10\sqrt{km}\kappa^2}$ suffices. This may be accomplished by Lemma 4.6.3. □

Lemma 4.6.2. *When the control inputs are chosen as $u_t = -\mathbb{K}x + \eta_t$, where $\eta_t \sim \{\pm 1\}^n$, it holds*

$$\|x_t\| \leq \sqrt{n}\kappa^3\gamma^{-1}W, \quad \|u_t\| \leq 2\sqrt{n}\kappa^4\gamma^{-1}W,$$
$$c_t(x_t, u_t) - \min_{K\in\mathcal{K}} c_t(x^K, u^K) \leq 16Gn\kappa^8\gamma^{-2}W^2.$$

Proof. In conjuction with Equation 4.5.1, the strong stability of $\mathbb{K}$ suffices to establish this claim.

$$\|x_t\| \leq (W + \|B\|\sqrt{n})\|Q\|\|Q^{-1}\|\sum_{i=0}^{t-1}\|L^i\| \leq (W + \kappa\sqrt{n})\kappa^2\sum_{i=0}^{t-1}(1-\gamma)^i$$

In addition to sub-multiplicativity of the norm, we use that $\sum_{i=0}^{t-1}(1-\gamma)^i \leq \gamma^{-1}$. □

4.6.1 Recovering moments

The following lemma promises an approximate recovery of $(A')^iB$'s through an appeal to arguments involving measures of concentration.

Lemma 4.6.3. *Algorithm 6 satisfies for all $j \in [k]$, with probability $1 - \delta$ or more*

$$\|N_j - (A')^j B\| \leq n\kappa^3\gamma^{-1}W\sqrt{\frac{8\log(mnk\delta^{-1})}{T_0 - k}}.$$

Proof. Let $N_{j,t} = x_{t+j+1}\eta_t^\top$. With Equation 4.5.1, the fact that η_i is zero-mean with isotropic unit covariance, and that it is chosen independently of $\{w_j\}, \{\eta_j\}_{j\neq i}$ implies $\mathbb{E}[N_j] = \mathbb{E}[N_{j,t}] = (A')^j B$.

$N_{j,t}$'s are bounded as $\|N_{j,t}\| \leq n\kappa^3\gamma^{-1}W$. Involving the same instance of η in multiple terms, they may not be independent across the second index t. To remedy this, define a sequence of centered random variables $\tilde{N}_{j,t} = N_{j,t} - (A')^j B$, and observe that they forms a martingale difference sequence with respect to $\{\eta_t\}$, ie.

$$\mathbb{E}[\tilde{N}_{j,t}|\eta_0, \ldots \eta_{t-1}] = \mathbb{E}[x_{t+j+1}\eta_t^\top|\eta_0, \ldots \eta_{t-1}] - (A')^j B = 0.$$

Together with a union bound on choice of $j \in [k]$, the application of Matrix Azuma inequality on a standard symmetric dilation of the sequence concludes the proof of the claim. □

4.6.2 Recovering system matrices

The following is a standard result on the perturbation analysis for linear systems (See [Bha13], for example). A proof is presented here for completeness.

Lemma 4.6.4. *Let x^* be the solution to the linear system $Ax = b$, and $\hat{x}$ be the solution to $(A + \Delta A)x = b + \Delta b$, then as long as $\|\Delta A\| \leq \sigma_{\min}(A)$, it is true that*

$$\|x^* - \hat{x}\| \leq \frac{\|\Delta b\| + \|\Delta A\|\|x^*\|}{\sigma_{min}(A) - \|\Delta A\|}$$

Proof. Observe the (in)equalities that imply the claim.

$$(A + \Delta A)(\hat{x} - x^*) = b + \Delta b - Ax^* - \Delta A x^* = \Delta b - \Delta A x^*$$
$$\|(A + \Delta A)^{-1}\| \leq \frac{1}{\sigma_{min}(A) - \|\Delta A\|}$$

□

Part II

Applications of Nonstochastic Control

Chapter 5

Regret Minimization in Iterative Learning Control

In this chapter, we consider the setting of iterative learning control, or model-based policy learning in the presence of uncertain, time-varying dynamics. In this setting, we propose a new performance metric, *planning regret*, which replaces the standard stochastic uncertainty assumptions with worst case regret. Based on recent advances in non-stochastic control, we design a new iterative algorithm for minimizing planning regret that is more robust to model mismatch and uncertainty. We provide theoretical and empirical evidence that the proposed algorithm outperforms existing methods on several benchmarks.

5.1 Introduction

Consider a robotic system learning to perform a novel task, e.g., a quadrotor learning to fly to a specified goal, a manipulator learning to grasp a new object, or a fixed-wing airplane learning to perform a new maneuver. We are particularly interested in settings where (i) the task requires one to *plan* over a given time horizon, (ii) we have access to an *inaccurate* model of the world (e.g., due to unpredictable external disturbances such as wind gusts or misspecification of physical parameters such as masses, inertias, and friction coefficients), and (iii) the robot is allowed to iteratively refine its control policy via multiple executions (i.e., rollouts) on the real world. Motivated by applications where real-world rollouts are expensive and time-consuming, our goal in this chapter is to learn to perform the given task as rapidly as possible. More precisely, given a cost function that specifies the

task, our goal is to learn a low-cost control policy using a small number of rollouts.

The problem described above is challenging due to a number of factors. The primary challenge we focus on in this chapter is the existence of unmodeled deviations from nominal dynamics, and external disturbances acting on the system. Such disturbances may either be random or potentially even adversarial. In this chapter, we adopt a *regret minimization* approach coupled with a recent paradigm called non-stochastic control to tackle this problem in generality. Specifically, consider a time-varying dynamical system given by the equation

$$x_{t+1} = f_t(x_t, u_t) + w_t, \tag{5.1.1}$$

where x_t is the state, u_t is the control input, and w_t is an arbitrary disturbance at time t. Given a horizon T, the performance of a control algorithm $\mathcal{A}$ may be judged via the aggregate cost it suffers on a cost function sequence $c_1, \dots c_T$ along its state-action trajectory $(x_1^{\mathcal{A}}, u_1^{\mathcal{A}}, \dots)$:

$$J(\mathcal{A}|w_{1:T}) = \frac{1}{T}\sum_{t=1}^{T} c_t\left(x_t^{\mathcal{A}}, u_t^{\mathcal{A}}\right).$$

For deterministic systems, an optimal open-loop control sequence $u_1 \dots u_T$ can be chosen to minimize the cost sequence. The presence of unanticipated disturbances often necessitates the superposition of a *closed-loop* correction policy π to obtain meaningful performance. Such closed-loop policies can modify the open-loop control sequence $u_1 \dots u_T$ to $u'_t = \pi(u_{1:t}, x_{1:t})$ which is a function of the observed history till time t, and facilitate adaptation to realized disturbances. To capture this, we define a comparative performance metric, which we call **Planning Regret**. In an episodic setting, for every episode i, an algorithm $\mathcal{A}$ adaptively selects control inputs while the rollout is performed under the influence of an arbitrary disturbance sequence $w^i_{1:T}$. Planning regret is the difference between the total cost of the algorithm's actions and that of the retrospectively optimal open-loop plan coupled with episode-specific optimal closed-loop policies (from a policy class Π). Regret, therefore, is the relative cost of not knowing the to-be realized disturbances in advance. Formally for a total of N rollouts, each of horizon T, it is defined as:

$$\textbf{Planning Regret} = \sum_{i=1}^{N} J(\mathcal{A}|w^i_{1:T}) - \min_{u^\star_{1:T}} \sum_{i=1}^{N} \min_{\pi^\star_i \in \Pi} J(u^\star_{1:T}, \pi^\star_i | w^t_{i,1:T}) \tag{5.1.2}$$

The motivation for our performance metric arises from the setting of Iterative Learning Control (ILC), where one assumes access to an imperfect (differentiable) simulator of real-world dynamics as

well as access to a limited number of rollouts in the real world. In such a setting the disturbances capture the model-mismatch between the simulator and the real-world. The main novelty in our formulation is the fact that, under vanishing regret, the closed-loop behavior of $\mathcal{A}$ is almost *instance-wise optimal* on the specific trajectory, and therefore adapts to the passive controls, dynamics and disturbance for each particular rollout. Indeed, worst-case regret is a stronger metric of performance than commonly considered in the planning/learning for control literature.

Our main result is an efficient algorithm that guarantees vanishing average planning regret for non-stationary linear systems and disturbance-action policies. We experimentally demonstrate that the algorithm yields substantial improvements over ILC in linear and non-linear control settings.

We present the relevant definitions including the setting in Section 5.2. The algorithm and the formal statement of the main result can be found in Section 5.3. In Section 5.4 we provide an overview of the algorithm and the proof via the proposal of a more general and abstract *nested online convex optimization (OCO) game*. This formulation can be of independent interest. Finally in Section 7.6, we provide the results and details of the experiments.

5.1.1 Related work

The literature on planning and learning in partially known MDPs is vast, and we focus here on the setting with the following characteristics:

1. We consider *model-aided* learning, which is suitable for situations in which the learner has some information about the dynamics, i.e. the mapping f_t in Equation 5.1.1, but not the disturbances w_t. We further assume that we can differentiate through the model. This enables efficient gradient-based algorithms.

2. We focus on the task of learning an episodic-agnostic control sequence, rather than a policy. This is aligned with the Pontryagin optimality principle [PBGM62, Ros15], and differs from dynamic programming approaches [SB18].

3. We accomodate arbitrary disturbance processes, and choose regret as a performance metric. This is a significant deviation from the literature on optimal and robust control [ZDG96a, Ste94b], and follows the lead of the recent paradigm of non-stochastic control [ABH$^+$19, HKS20, SSH20].

4. Our approach leverages multiple real-world rollouts. This access model is most similar to the iterative learning control (ILC) paradigm [OH05, ACM07]. For comparison, the model-predictive control (MPC) paradigm allows for only one real-world rollout on which performance

is measured, and all other learning is permitted via access to a simulator.

Optimal, Robust and Online Control. Classic results [Ber05, ZDG96a, Ted20] in optimal control characterize the optimal policy for linear systems subject to i.i.d. perturbations given explicit knowledge of the system in advance. Beyond stochastic perturbations, robust control approaches [ZD98] compute the best controller under worst-case noise.

Recent work in machine learning [AYS11, DMM$^+$18, MTR19, CHK$^+$18, AHS19] study regret bounds vs. the best linear controller in hindsight for online control with known and unknown linear dynamical systems. Online control was extended to adversarial perturbations, giving rise to the nonstochastic control model. In this general setting regret bounds were obtained for known/unknown systems as well as partial observation [ABH$^+$19, HKS20, SSH20, Sim20].

Planning with inaccurate models. Model predictive control (MPC) [May14] provides a general scheme for planning with inaccurate models. MPC operates by applying model-based planning, (eg. iLQR [LT04, TL05]), in a receding-horizon manner. MPC can also be extended to robust versions [BM99, MSR05, LCRM04, SZG$^+$21] that explicitly reason about the parametric uncertainty or external disturbances in the model. Recently, MPC has also been viewed from the lens of online learning [WCSB19]. The setting we consider here is more general than MPC, allowing for iterative policy improvement across *multiple rollouts* on the real world.

An adjacent line of work on *learning MPC* [HWMZ20, RB17] focuses on constraint satisfaction and safety considerations while learning models simultaneously with policy execution.

Iterative Learning Control (ILC). ILC is a popular approach for tackling the setting considered. ILC operates by iteratively constructing a policy using an inaccurate model, executing this policy on the real world, and refining the policy based on the real-world rollout. ILC can be extended to use real-world rollouts to update the model (see, e.g., [AQN06]). For further details regarding ILC, we refer the reader to the text [Moo12]. Robust versions of ILC have also been developed in the control theory literature [dR96], using H-infinity control to capture bounded disturbances or uncertainty in the model.

However, most of the work in robust control, typically account for *worst-case* deviations from the model and can lead to extremely conservative behavior. In contrast, here we leverage the recently-proposed framework of *non-stochastic control* to capture *instance-specific* disturbances. We demonstrate both empirically and theoretically that the resulting algorithm provides significant gains in terms of sample efficiency over the standard ILC approach.

Meta-Learning. Our setting, analysis and, in particular, the nested OCO setup bears similarity to formulations for gradient-based meta-learning (see [FAL17] and references therein). In particular, as we detail below, the nested OCO setting we consider is a generalization of the setting considered in [BKT19]. We further detail certain improvements/advantages our algorithm and analysis provides over the results in [BKT19]. We believe this connection with Meta-Learning to be of independent interest.

Here, we describe how the nested-OCO formulation proposed in the chapter can be used to derive upto a small constant factor, the gradient based meta-learning results presented in [BKT19] by reducing their setting to the nested-OCO setting and applying Algorithm 9. The reduction requires setting the x, y space, i.e. $\mathcal{K}_1, \mathcal{K}_2$ to be $\Theta \subseteq \mathcal{R}^d$ and a ball in $\mathcal{R}^d$ of diameter D^* (according to the notation in [BKT19]). Further, we set the function $f_t^i(x, y) \triangleq l_{t,i}(x + y)$. The reduction recovers the same guarantee as the result in [BKT19] upto a factor of 2.

We note that [BKT19] provide an algorithm that works without the knowledge of D^*, but such an extension is standard in OCO literature and can be handled similarly to [BKT19]. Further we acknowledge that for the particular problem considered in [BKT19], the constant factor is important as a straightforward algorithm also achieves the same rate if constant factors are ignored, a fact highlighted in the original chapter. On the other hand, our formulation allows for a stronger comparator even in the [BKT19] setup.

We would like to highlight that our nested-OCO setup allowing for different x, y spaces is more general than the setup typically considered in initialization-based meta-learning. Owing to this generality, the algorithm we provide naturally performs a gradient step on the true function value for the outer loop as opposed to a distance based function as in [BKT19]. Further exploring the effectiveness of our algorithm for meta-learning is left as interesting future work.

5.2 Problem setting

5.2.1 Notation

The norm $\|\cdot\|$ refers to the ℓ_2 norm for vectors and spectral norm for matrices. For any natural number n, the set $[n]$ refers to the set $\{1, 2 \dots n\}$. We use the notation $v_{a:b} \triangleq \{v_a \dots v_b\}$ to denote a sequence of vectors/matrices. Given a set S, we use $v_{a:b} \in S$ to represent element wise inclusion, i.e. $\forall j \in [a, b], v_j \in S$; $\text{Proj}_S(v_{a:b})$ represents the element-wise ℓ_2 projection onto to the set S. $v_{a:b,c:d}$ denotes a sequence of sequences, i.e. $v_{a:b,c:d} = \{v_{a,c:d} \dots v_{b,c:d}\}$ with $v_{a,c:d} = \{v_{a,c} \dots v_{a,d}\}$.

5.2.2 Basic definitions

A **dynamical system** is specified via a start state $x_0 \in \mathbb{R}^{d_x}$, a time horizon T and a sequence of transition functions $f_{1:T} = \{f_t | f_t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \to \mathbb{R}^n\}$. The system produces a T-length sequence of states $(x_1, \dots x_{T+1})$ when subject to an T-length sequence of actions $(u_1 \dots u_T)$ and disturbances $\{w_1, \dots w_T\}$ according to the following dynamical equation[1]

$$x_{t+1} = f_t(x_t, u_t) + w_t.$$

Through the chapter the only assumption we make about the disturbance w_t is that it is supported on a set of bounded diameter W. We assume full observation of the system, i.e. the states x_t are visible to the controller. We also assume the passive transition function to be **known** beforehand. These assumptions imply that we fully observe the instantiation of the disturbances $w_{1:T}$ during runs of the system.

The actions above may be adaptively chosen based on the observed state sequence, ie. $u_t = \pi_t(x_1, \dots x_t)$ for some non-stationary policy $\pi_{1:T} = \{\pi_1, \dots \pi_T\}$. We consider the policy to be deterministic (a restriction made for convenience). Therefore the state-action sequence $\{x_t, u_t\}_{t=1}^T$, defined as $x_{t+1} = f_t(x_t, u_t) + w_t, u_t = \pi_t(x_1 \dots x_t)$, thus produced is a sequence determined by $w_{1:T}$, fixing the policy, and the system.

A ***rollout*** *of horizon* $\mathbf{T}$ *on* $f_{1:T}$ refers to an evaluation of the above sequence for T time steps. When the dynamical system will be clear from the context, for the rest of the chapter, we drop it from our notation. Given a cost function sequence $\{c_t\} : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ the ***loss*** of executing a policy π on the dynamical system f with a particular disturbance sequence given by $w_{1:T}$ is defined as

$$J(\pi_{1:T} | f_{1:T}, w_{1:T}) \triangleq \frac{1}{T}\left[\sum_{\tau=1}^{T} c_t(x_t, u_t)\right].$$

Assumption 5.2.1. *We will assume that the cost c_t is a twice differentiable convex function and that the value, gradient and hessian of the cost function c_t is available. Further we assume,*

- ***Lipschitzness:*** *There exists a constant G such that if $\|x\|, \|u\| \leq D$ for some $D > 0$, then $\|\nabla_x c_t(x, u)\|, \|\nabla_u c_t(x, u)\| \leq GD$.*
- ***Smoothness:*** *There exists a constant β such that for all x, u, $\nabla^2 c_t(x, u) \preceq \beta I$.*

When the dynamical system and the noise sequence are clear from the context we suppress them

[1] For the sake of simplicity, we do not consider a terminal cost, and consequently drop the last state from the description.

from the notation for the cost denoting it by $J(\pi_{1:T})$. A particular sub-case which will be of special interest to us is the case of linear dynamical systems (LDS). Formally, a (non-stationary) linear dynamical system is described by a sequence of matrices $AB_{1:T} = \{(A_t, B_t) \in \mathbb{R}^{d_x, d_x} \times \mathbb{R}^{d_x, d_u}\}_{t=1}^{T}$ and the transition function is defined as $x_{t+1} = A_t x_t + B_t u_t$.

Assumption 5.2.2. *We will assume that the linear dynamical system $AB_{1:T}$ is (κ, δ)- strongly stable for some $\kappa > 0$ and $\delta \in (0, 1]$, i.e. for if every t, we have that $\|A_t\| \leq 1 - \delta, \|B_t\| \leq \kappa$.*

We note that all the results in the chapter can be easily generalized to a weaker notion of strong stability where the linear dynamical system is (κ, δ)- strongly stable if there exists a sequence of matrices $K_{1:T}$, such that for every t, we have that $\|A_t - B_t K_t\| \leq 1 - \delta, \|B_t\|, \|K_t\| \leq \kappa$. A system satisfying such an assumption can be easily transformed to a system satisfying Assumption 5.2.2 by setting $A_t = A_t - B_t K_t$. This redefinition is equivalent to appending the linear policy K_t on top of the policy being executed. While we present the results for the case when $K_T = 0$, the only difference the non-zero case makes to our analysis is potentially increasing the norm of the played actions which can still be shown to be bounded. Overall this nuance leads to a difference to our main result only in terms of factors polynomial in the system parameters. Hence for convenience, we state our results under Assumption 5.2.2. The assumption of strong-stability (in a weaker form as allowed by stationary systems) has been popular in recent works on online control [CHK$^+$18, ABH$^+$19] and the above notion generalizes it to non-stationary systems.

5.2.3 Policy classes

Open-Loop Policies. Given a convex set $\mathcal{U} \in \mathbb{R}^{d_u}$, consider a sequence of control actions, $u_{1:T} \in \mathcal{U}$. We define (by an overload of notation), the open-loop policy $u_{1:T}$ as a policy which plays at time t, the action u_t. The set of all such policies is defined as $\Pi_{\mathcal{U}} \triangleq \mathcal{U}^{\otimes T}$.

Given two policies we define the sum of the two (denoted by $\pi_1 + \pi_2$) as the policy for which the action at time t is the sum of the action recommended by policy π_1 and π_2.

Linear Policies. Given a matrix $K \in \mathbb{R}^{d_u, d_x}$, a *linear policy* [2] denoted (via an overload of notation) by K is a policy that plays action $u_t = K x_t$. Such linear state-feedback policies are known to be optimal for the LQR problem and for H_∞ control [ZDG96a].

[2] For notational simplicity, we do not include an affine offset c_t in the definition of our linear policy; this can be included with no change in results across the chapter.

Disturbance Action Controllers (DAC). A generalization of the class of linear policies can be obtained via the notion of disturbance-action policies (see [ABH+19]) defined as follows. A disturbance action policy $\pi_{M_{1:L}}$ of memory length L is defined by a sequence of matrices $M_{1:L} \triangleq \{M_1 \dots M_L\}$ where each $M_i \in \mathcal{M} \subseteq \{\mathbb{R}^{d_u \times d_x}\}$, with the action at time step t given by

$$[\pi_{M_{1:L}}]_t \triangleq \sum_{j=1}^{L} M_j w_{t-j}. \tag{5.2.1}$$

A natural class of matrices from which the above feedback matrices can be picked is given by fixing a number $\gamma > 0$ and picking matrices spectrally bounded by γ, i.e. $\mathcal{M}_\gamma \triangleq \{M | M \in \mathbb{R}^{d_u \times d_x}, \|M\| \leq \gamma\}$. We further overload the notation for a disturbance action policy to incorporate an open-loop control sequence $u_{1:T}$, defined as $\pi_{M_{1:L}}(u_{1:T}) \triangleq \{u_t + \sum_{j=1}^{L} M_j w_{t-j}\}_{t=1}^{T}$.

5.2.4 Planning regret with disturbance-action policies

As discussed, a natural idea to deal with adversarial process disturbance is to plan (potentially oblivious to it), producing a sequence of open loop ($u_{1:T}$) actions and appending an adaptive controller to *correct* for the disturbance online. However the disturbance in practice could have structure across rollouts, which can be leveraged to improve the plan ($u_{1:T}$), with the knowledge that we have access to an adaptive controller. To capture this, we define the notion of an online planning game and the associated notion of planning regret below.

Definition 5.2.3 (Online Planning)**.** *It is defined as an N round/rollout game between a player and an adversary, with each round defined as follows:*

- *At every round i the player given the knowledge of a new dynamical system $f^i_{1:T} = \{f^i_1 \dots f^i_T\}$, proposes a policy $\pi^i_{1:T} = \{\pi^i_1 \dots \pi^i_T\}$.*
- *The adversary then proposes a noise sequence $w^i_{1:T}$ and a cost sequence $c^i_{1:T}$.*
- *A rollout of policy $\pi^i_{1:T}$ is performed on the system $f^i_{1:T}$ with disturbances $w^i_{1:T}$ and the cost suffered by the player $J_i(\pi^i_{1:T}) \triangleq J(\pi^i_{1:T} | f^i_{1:T}, w^i_{1:T})$.*

The task of the controller is to minimize the cost suffered. We measure the performance of the controller via the following objective, defined as **Planning-Regret**, which measures the performance against the metric of producing the best-in-hindsight open-loop plan, having been guaranteed the optimal adaptive control policy for every single rollout. The notion of adaptive control policy we use is the disturbance-action policy class defined in Equation 5.2.1. In particular, they generalize linear

policies for stationary systems and lend convexity. Formally, planning regret is defined as follows:

$$\textbf{Planning Regret} = \sum_{i=1}^{N} J_i(\pi^i_{1:T}) - \min_{u_{1:T}} \sum_{i=1}^{N} \left(\min_{M_{1:L}} J_i\left(\pi_{M_{1:L}}(u_{1:T})\right) \right)$$

5.3 Main algorithm and result

In this section we propose the algorithm **iGPC** (Iterative Gradient Perturbation Controller; Algorithm 7) to minimize Planning Regret. The algorithm at every iteration given an open-loop policy $u_{1:T}$ performs a rollout overlaying an online DAC adaptive controller GPC (Algorithm 8). Further the base policy $u_{1:T}$ is updated by performing gradient descent (or any other local policy improvement) on u fixing the offsets suggested by GPC. We show the following guarantee on average planning regret for Algorithm 7 for linear dynamical systems.

Theorem 5.3.1. *Let $\mathcal{U} \subseteq \mathbb{R}^{d_u}$ be a bounded convex set with diameter U. Consider the online planning game (Definition 5.2.3) with linear dynamical systems $\{AB^i_{1:T}\}_{i=1}^N$ satisfying Assumption 5.2.2 and cost functions $\{c_{1:T}\}_{i=1}^N$ satisfying Assumption 5.2.1. Then we have that Algorithm 7 (when executed with appropriate parameters), for any sequence of disturbances $\{w^i_{1:T}\}_{i=1}^N$ with each $\|w^i_t\| \leq W$ and any $\gamma \geq 0$, produces a sequence of actions with planning regret bounded as*

$$\frac{1}{N}\left(\sum_{i=1}^{N} J_i(\pi^i_{1:T}) - \min_{u_{1:T} \in \mathcal{U}} \left(\sum_{i=1}^{N} \min_{M_{1:L} \in \mathcal{M}_\gamma} J_i\left(\pi_{M_{1:L}}(u_{1:T})\right) \right) \right) \leq \tilde{O}\left(\frac{1}{\sqrt{T}} + \frac{1}{\sqrt{N}} \right).$$

where $\mathcal{M}_\gamma = \{M | M \in \mathbb{R}^{d_u, d_x}, \|M\| \leq \gamma\}$.

The $\tilde{O}$ notation above subsumes factors polynomial in system parameters $\kappa, \gamma, \delta^{-1}, U, W, G$ and $\log(T)$.

5.4 Algorithmic details and proof sketch

In this section we provide an overview of the derivation of the algorithm and the proof for Theorem 5.3.1. We introduce an online learning setting that is the main building block of our algorithm. The setting applies more generally to control/planning and our formulation of planning regret in linear dynamical systems is a specification of this setting.

Algorithm 7 iGPC Algorithm

Inputs: [Online] $f_{1:T}^{1:N}$: Dynamical Systems, $w_{1:T}^{1:N}$: Disturbances , $c_{1:T}^{1:N}$
Parameters : $\mathcal{U}$, η_{out} : Learning Rate
Initialize $u_{1:T}^1 \in \mathcal{U}$ arbitrarily.
for $i = 1 \dots N$ **do**
  Receive a dynamical system $f_{1:T}^i$.
  Rollout the policy $u_{1:T}^i$ with GPC (Alg. 8),

$$\{x_{1:T}^i, a_{1:T}^i, w_{1:T}^i, o_{1:T}^i\} = \text{GPCRollout}(f_{1:T}^i, u_{1:T}^i).$$

  Update: Compute the update to the policy,

$$\nabla_i = \nabla_{u_{1:T}} J(u_{1:T}^i + o_{1:T}^i | f_{1:T}^i, w_{1:T}^i)$$
$$u_{1:T}^{i+1} = \text{Proj}_{\mathcal{U}} \left(u_{1:T}^i - \eta_{\text{out}} \nabla_i \right).$$

end for

Algorithm 8 GPCRollout

Inputs: $f_{1:T}$: dynamical system, $u_{1:T}$: input policy, [Online] $w_{1:T}$: disturbances, $c_{1:T}$: costs.
Parameters L : Window, η_{in} : Learning rate, γ : Feedback bound, S : Lookback
Initialize $M_{1,1:L} = \{M_{1,j}\}_{j=1}^{L} \in \mathcal{M}_\gamma$.
Set $w_i = 0$ for all $i \leq 0$.
for $t = 1 \dots T$ **do**
  Compute Offset: $o_t = \sum_{r=1}^{L} M_{t,r} \cdot w_{t-r}$.
  Play action: $a_t = u_t + o_t$.
  Suffer Cost: $c_t(x_t, a_t)$
  Observe state: x_{t+1}.
  Compute perturbation:

$$w_t = x_{t+1} - f_t(x_t, a_t).$$

  Do a gradient step on the GPCLoss Equation 5.4.1

$$M_{t+1,1:L} = \text{Proj}_{\mathcal{M}_\gamma} \left(M_{t,1:L} - \eta_{\text{in}} \nabla \text{GPCLoss}(arg) \right),$$

  where arg captures policy $M_{t,1:L}$, open-loop plan $u_{t-S+1:t}$, disturbances $w_{t-S-L+1:t-1}$, transition $f_{t-S+1,t-1}$, cost c_t in Equation 5.4.1 and gradient is taken with respect to the M parameter.
end for
Return $x_{1:T}, a_{1:T}, w_{1:T}$, $o_{1:T}$.

5.4.1 Nested OCO and planning regret

Setting: Consider an online convex optimization(OCO) problem [Haz16], where the iterations have a nested structure, divided into inner and outer iterations. Fix two convex sets $\mathcal{K}_1$ and $\mathcal{K}_2$. After every one out of N outer iterations, the player chooses a point $x_i \in \mathcal{K}_1$. After that there is a sequence of T inner iterations, where the player chooses $y_t^i \in \mathcal{K}_2$ at every iteration. After this choice, the adversary chooses a convex cost function $f_t^i \in \mathcal{F} \subseteq \mathcal{K}_1 \times \mathcal{K}_2 \to \mathbb{R}$, and the player suffers a cost of

$f_t^i(x_i, y_t^i)$. The goal of the player is to minimize Planning Regret:

$$\textbf{Planning Regret} = \sum_{i=1}^{N}\sum_{t=1}^{T}\frac{f_t^i(x_i, y_t^i)}{T} - \min_{x^\star\in\mathcal{K}_1}\sum_{i=1}^{N}\min_{y^\star\in\mathcal{K}_2}\sum_{t=1}^{T}\frac{f_t^i(x^\star, y^\star)}{T}$$

To state a general result, we assume access to two online learners denoted by $\mathcal{A}_1, \mathcal{A}_2$, that are guaranteed to provide sub-linear regret bounds over *linear* cost functions on the sets $\mathcal{K}_1, \mathcal{K}_2$ respectively in the standard OCO model. We denote the corresponding regrets achieved by $R_N(\mathcal{A}_1), R_T(\mathcal{A}_2)$. A canonical algorithm for online linear optimization (OLO) is online gradient descent [Zin03], which is what we use in the sequel. The theory presented here applies more generally. [3] Algorithm 9 lays out a general algorithm for the Nested-OCO setup.

Algorithm 9 Nested-OCO Algorithm

Inputs: Algorithms $\mathcal{A}_1, \mathcal{A}_2$.
Initialize $x_1 \in \mathcal{K}_1$ arbitrarily.
for $i = 1 \dots N$ **do**
  Initialize $y_0^i \in \mathcal{K}_2$ arbitrarily.
  for $t = 1 \dots T$ **do**
    Define loss function over $\mathcal{K}_2$ as

$$h_t^i(y) \triangleq \nabla_y f_t^i(x_i, y_t^i)\cdot y.$$

    Update $y_{t+1} \leftarrow \mathcal{A}_2(h_0^i \dots h_t^i)$.
  end for
  Define loss function over $\mathcal{K}_1$ as

$$g_i(x) \triangleq \sum_{t=1}^{T}\nabla_x f_t^i(x_i, y_t^i)\cdot x.$$

  Update $x_{s+1} \leftarrow \mathcal{A}_1(g_1, ..., g_i)$.
end for

Theorem 5.4.1. *Algorithm 9 with sub-algorithms $\mathcal{A}_1, \mathcal{A}_2$ with regrets $R_N(\mathcal{A}_1), R_T(\mathcal{A}_2)$ ensures the following regret guarantee on the average planning regret,*

$$\frac{\text{PlanningRegret}}{N} \le \frac{R_N(\mathcal{A}_1)}{N} + \frac{R_T(\mathcal{A}_2)}{T}.$$

When using Online Gradient Descent as the base algorithm, the average regret scales as $O\left(\frac{1}{\sqrt{N}} + \frac{1}{\sqrt{T}}\right)$.

[3] Regret for OLO depends on function bounds, which correspond to gradient bounds here. For clarity we omit this dependence from the notation for regret.

Proof of Theorem 5.4.1. Let $x^\star \in \mathcal{K}_1$ be any point and let $y^\star_{1:T} \in \mathcal{K}_2$ be any sequence. We have

$$\begin{aligned}
&\frac{\sum_{i=1}^N \sum_{t=1}^T f_t^i(x_i, y_t^i) - f_t^i(x^\star, y_i^\star)}{TN} \\
\leq &\frac{\sum_{i=1}^N \sum_{t=1}^T \nabla_x f_t^i(x_i - x^\star)}{TN} \\
&\qquad + \frac{\sum_{i=1}^N \sum_{t=1}^T \nabla_y f_t^i(y_t^i - y^\star)}{TN} \\
= &\frac{\sum_{i=1}^N [g_i(x_i) - g_i(x^\star)]}{TN} \\
&\qquad + \frac{\sum_{i=1}^N \sum_{t=1}^T [h_t^i(y_t) - h_t^i(y_i^\star)]}{TN} \\
\leq &\frac{R_N(\mathcal{A}_1)}{N} + \frac{\cdot R_T(\mathcal{A}_2)}{T},
\end{aligned}$$

where the first inequality follows by convexity and the last inequality follows by the regret guarantees and noting that the functions g_i are naturally scaled up by a factor of T. □

5.4.2 Proof sketch for Theorem 5.3.1

The main idea behind the proof is to reduce to the setting of Theorem 5.4.1. In the reduction the x variable corresponds to the open loop controls $u_{1:T} \in \mathcal{U}$ and the variables y_t^i correspond to the closed-loop disturbance-action policy $M_{t,1:L}^i \in \mathcal{M}_\gamma$. The algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ are instantiated as Online Gradient Descent with appropriately chosen learning rates.

We begin the reduction by using the observation in [ABH$^+$19] that costs are convex with respect to the variables u, M, for *linear dynamical systems* with convex costs. With convexity, prima-facie the reduction seems immediate, however this is impeded by the counterfactual notion of policy regret which implies that cost at any time is dependent on previous actions. This nuance in the reduction from Theorem 5.4.1 is only applicable to the closed loop policies M, the open loop part $u_{1:T}$) on the other hand, follows according to the reduction and hence direct OGD is applied (Line 6, Algorithm 7).

To resolve the issue of the counterfactual dependence, we use the techniques introduced in the OCO with memory framework proposed by [AHM15] and recently employed in the work of [ABH$^+$19]. We leverage the underlying stability of the dynamical system to ensure that cost at time t depends only on a bounded number of previous rounds, say S. We then define a proxy loss denoted by GPCLoss, corresponding to the cost incurred by a stationary closed-loop policy executing for the previous S time steps. Formally, given a dynamical system $f_{1:S}$, perturbations $w_{1:S}$, a cost function c, a non-stationary open-loop policy $u_{1:S}$, GPCLoss is a function of closed-loop transfer $M_{1:L}$ defined

as follows. Consider the following iterations with $y_1 = 0$,

$$\begin{aligned} a_j &\triangleq u_j + \sum_{r=1}^{L} M_r w_{j-r}, \\ y_j &\triangleq f_{j-1}(y_{j-1}, a_{j-1}) + w_{j-1} \quad \forall j \in [1, S], \\ \text{GPCLoss}(M_{1:L}, u_{1:S}, w_{-L+1:S-1}, f_{1:S-1}, c) &\triangleq c(y_S, a_S). \end{aligned} \tag{5.4.1}$$

The algorithm updates by performing a gradient descent step on this loss, i.e. $M^i_{t+1,1:L} = M^i_{t,1:L} - \eta \nabla_M \text{GPCLoss}(\cdot)$. The proof proceeds by showing that the actual cost and its gradient is closely tracked by their proxy GPC Loss counterparts with the difference proportional to the learning rate (Lemma 5.5.4). Choosing the learning rate appropriately then completes the proof.

5.5 Restatement of the main theorem and proof

We provide the following restatement of Theorem 5.3.1 with details regarding the parameters and the dependence on the system parameters. To state the results concisely, we assume that all the appropriate assumed constants, i.e. $\kappa, \gamma, G, \beta, U, W$ are greater than 1. This is done to upper bound the sum of two constants by twice their product. All the results hold by replacing any of these constants by the max of the constant and 1.

Theorem 5.5.1. *Let $\mathcal{U} \subseteq \mathbb{R}^{d_u}$ be a bounded convex set with diameter U. Consider the online planning game(Definition 5.2.3) with linear dynamical systems $\{AB^i_{1:T}\}_{i=1}^N$ satisfying Assumption 5.2.2 and cost functions $\{c_{1:T}\}_{i=1}^N$ satisfying Assumption 5.2.1. Then we have that Algorithm 7(when executed with appropriate parameters), for any sequence of disturbances $\{w^i_{1:T}\}_{i=1}^N$ with each $\|w^i_t\| \leq W$ and any $\gamma \geq 0$, produces a sequence of actions with planning regret bounded as*

$$\frac{1}{N}\left(\sum_{i=1}^{N} J_i(\pi^i_{1:T}) - \min_{u_{1:T} \in \mathcal{U}} \left(\sum_{i=1}^{N} \min_{M_{1:L} \in \mathcal{M}_\gamma} J_i\left(\pi_{M_{1:L}}(u_{1:T})\right)\right)\right) \leq \left(\frac{c_{\text{in}} \log^2(T)}{\sqrt{T}} + \frac{c_{\text{out}}}{\sqrt{N}}\right).$$

where $\mathcal{M}_\gamma = \{M | M \in \mathbb{R}^{d_u, d_x}, \|M\| \leq \gamma\}$ and $c_{\text{in}}, c_{\text{out}}$ are constants depending on system parameters as follows

$$c_{\text{out}} = \tilde{O}\left(GU(U + \gamma LW)\kappa^2 \delta^{-2}\right)$$

$$c_{\text{in}} = \tilde{O}\left(\sqrt{\gamma^3 \kappa^4 \delta^{-3} \beta G^2 L^5 W^3 (U + \gamma LW)^2}\right).$$

Here $\tilde{O}$ subsumes constant factors and factors poly-logarithimic in the the arguments of $\tilde{O}$. To achieve the above bound, Algorithm 7 is to be executed with parameters, learning rate $\eta_{\text{out}} = \frac{U}{G\kappa\delta^{-2}(\kappa U+\kappa\gamma LW+W)\sqrt{N}}$, with the inner execution of Algorithm 8 is performed with parameters $\eta_{\text{in}} = \frac{\gamma^2 L^2}{\sqrt{12\gamma\kappa^4\delta^{-5}\beta G^2 L^3 W^3 (U+\gamma LW)^2}}$ and $S = \delta^{-1}\log(\eta_{\text{in}})$.

5.5.1 Requisite definitions

Before proving the theorem we set up some useful definitions. Fix a linear dynamical system $AB_{1:T}$ and a disturbance sequence $w_{1:T}$. For any sequence $u_{1:T} \in \mathcal{U}$ and $M_{1:T,1:L} \in \mathcal{M}_\gamma$, we define T functions $x_{1:T}(\cdot|AB_{1:T}, w_{1:T}), a_{1:T}(\cdot|AB_{1:T}, w_{1:T})$, denoting the action played and the state visited at time t upon execution of the policies together. Herein we drop $AB_{1:T}, w_{1:T}$ from the notation when clear from the context. Formally, consider the following definitions for all t,

$$a_t(u_{1:T}, M_{1:T,1:L}) \triangleq u_t + \sum_{r=1}^{L} M_{t,r} w_{t-r} \tag{5.5.1}$$

$$x_1(u_{1:T}, M_{1:T,1:L}) \triangleq 0 \qquad x_{t+1}(u_{1:T}, M_{1:T,1:L}) \triangleq A_t x_t(u_{1:T}, M_{1:T,1:L}) + B_t a_t + w_t \tag{5.5.2}$$

Given a sequence of cost functions $c_{1:T}(x,u) : \mathbb{R}^{d_x \times d_u} \to \mathbb{R}$, satisfying Assumption 5.2.1, define via an overload of notation, the cost functions c_t as a function of $u_{1:T}, M_{1:T,1:L}$ as follows

$$\forall t \in [1:T], \qquad c_t(u_{1:t}, M_{1:T,1:L}) = c_t(x_t(u_{1:t}, M_{1:T,1:L}), a_t(u_{1:T}, M_{1:T,1:L})) \tag{5.5.3}$$

Naturally, according to our definition of the total cost J of the rollout we get that

$$J(u_{1:T}, M_{1:T,1:L}) = \frac{1}{T}\sum_{t=1}^{T} c_t(u_{1:t}, M_{1:T,1:L})$$

Next, we expand upon the recursive definition of $x_t(\cdot,\cdot)$ via the following operators,

Definition 5.5.2. *Given a linear dynamical system $AB_{1:T}$, define the following transfer matrices*

$$\forall j \in [T], \forall k \in [j+1, T] \quad T_{j\to k} \in \mathbb{R}^{d_x \times d_u} \qquad T_{j\to k} \triangleq \begin{cases} I & \text{if } k = j+1 \\ \left(\Pi_{t=j+2}^{k} A_t\right) & \text{otherwise} \end{cases}$$

Additionally given a disturbance sequence $w_{1:T}$, define the following linear operator over matrix sequences $M_{1:T,1:L}$

$$\forall j \in [T], \forall k \in [j+1, T] \quad \psi_{j\to k}^{M} : [\mathbb{R}^{d_u \times d_x}]^{T \times L} \to \mathbb{R}^{d_x}$$

$$\psi^M_{j\to k}(M_{1:T,1:L}) = \sum_{t=j}^{k-1}\left(T_{t\to k}B_t\left(\sum_{r=1}^{L} M_{t,r}w_{k-r}\right)\right)$$

It can be observed via unrolling the recursion and the definitions above that

$$x_t(u_{1:T}, M_{1:T,1:L}) = \sum_{j=1}^{t-1} T_{j\to t}(B_j u_j + w_j) + \psi^M_{1\to t}(M_{1:T,1:L}). \tag{5.5.4}$$

Since x_t, a_t are linear functions of $u_{1:T}, M_{1:T,1:L}$, therefore we have that $c_t(u_{1:T}, M_{1:T,1:L})$ is a convex function of its arguments. The next lemma further shows that the gradient of the total cost with respect to the argument $u_{1:T}$ is bounded, as stated in the following lemma.

Lemma 5.5.3. *Given a linear system $AB_{1:T}$ satisfying Assumption 5.2.2, a bounded disturbance sequence $w_{1:T}$ and a cost sequence c_t satisfying Assumption 5.2.1, then for any $\gamma \geq 0$,$\mathcal{U}$, let $u_{1:T} \in \mathcal{U}, M_{1:T,1:L} \in \mathcal{M}_\gamma$ be two sequences, then we have that*

$$\left\|\nabla_{u_j}\left(\sum_{t=1}^{T} c_t(u_{1:T}, M_{1:T,1:L})\right)\right\| \leq 2G\kappa\delta^{-2}(\kappa U + \kappa\gamma LW + W)$$

We provide the proof of the lemma further in the section. Using the lemma we are now ready to prove Theorem 5.3.1.

Proof of Theorem 5.3.1. Lets fix a particular rollout i. Let $AB^i_{1:T}$ be the dynamical system and $w^i_{1:T}$ be the disturbance supplied. Further $u^i_{1:T}$ be the open loop control sequence played at round i and $M^i_{1:T,1:L}$ be the disturbance feedback sequence played by the GPC subroutine. By definition we have that the state achieved

$$x^i_t = x_t(u^i_{1:T}, M^i_{1:T,1:L}) \qquad a^i_t = a_t(u^i_{1:T}, M^i_{1:T,1:L})$$

We have for convenience dropped the system and disturbance from our notation. The total cost at round i incurred by the algorithm by definition is

$$J = \sum_{i=1}^{N}\frac{1}{T}\left(\sum_{t=1}^{T} c^i_t(u^i_{1:T}, M^i_{1:T,1:L})\right)$$

Fix the sequence of comparators to be $\overset{*}{u}_{1:T}, \{\overset{*}{M}{}^i_{1:L}\}_{i=1}^N$. The comparator cost by definition then is

$$\overset{*}{J} = \sum_{i=1}^{N}\frac{1}{T}\left(\sum_{t=1}^{T} c^i_t(\overset{*}{u}_{1:T}, \mathcal{T}_T\overset{*}{M}{}^i_{1:L})\right),$$

where given a sequence $v_{a:b}$, we define the tiling operator $\mathcal{T}_k$, which creates a nested sequence of outer length k by tiling with copies of the sequence $v_{a:b}$, i.e. $\mathcal{T}_k v_{a:b} = [v_{a:b}, v_{a:b} \dots v_{a:b}]$. We therefore have the following calculation for the regret which follows from the convexity of the cost function c_t with respect to u, M as established before,

$$\begin{aligned}
&\sum_{i=1}^{N}\sum_{t=1}^{T}\left(c_t^i(u_{1:T}^i, M_{1:T,1:L}^i) - c_t^i(\overset{*}{u}_{1:T}, \mathcal{T}_T \overset{*}{M}{}_{1:L}^i)\right) \\
&\leq \sum_{i=1}^{N}\sum_{t=1}^{T}\left(\nabla_u c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)(u_{1:T}^i - \overset{*}{u}_{1:T}) + \nabla_M c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)(M_{1:T,1:L}^i - \overset{*}{M}{}_{1:L}^i)\right) \\
&= \underbrace{\sum_{i=1}^{N}\sum_{t=1}^{T}\left(\nabla_u c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)(u_{1:T}^i - \overset{*}{u}_{1:T})\right)}_{\text{Outer Regret}} + \underbrace{\sum_{i=1}^{N}\sum_{t=1}^{T}\left(\nabla_M c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)(M_{1:T,1:L}^i - \overset{*}{M}{}_{1:L}^i)\right)}_{\text{Inner Regret}}
\end{aligned}$$

We analyze the both the terms above separately. We begin by analyzing the first term.

Outer Regret: Consider the following calculation

$$\sum_{i=1}^{N}\sum_{t=1}^{T}\left(\nabla_u c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)(u_{1:T}^i - \overset{*}{u}_{1:T})\right) = \sum_{j=1}^{T}\sum_{i=1}^{N}\underbrace{\nabla_{u_j}\left(\sum_{t=1}^{T} c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)\right)}_{\triangleq g_{ij}^u}(u_j^i - \overset{*}{u}_j).$$

Note that by definition of the algorithm, we have that for all i, j

$$u_j^{i+1} = \text{Proj}_{\mathcal{U}}(u_j^i - \eta_{\text{out}} g_{ij}^u),$$

which via the pythagorean inequality implies that

$$\|u_j^{i+1} - \overset{*}{u}_j\|^2 \leq \|u_j^i - \eta_{\text{out}} g_{ij}^u - \overset{*}{u}_j\|^2$$

Combining the above equations we immediately get that

$$\begin{aligned}
\sum_{i=1}^{N}\sum_{t=1}^{T}\left(\nabla_u c_t^i(u_{1:T}^i, M_{1:T,1:L}^i)(u_{1:T}^i - \overset{*}{u}_{1:T})\right) &\leq \sum_{j=1}^{T}\sum_{i=1}^{N}\frac{1}{2}\left(\eta_{\text{out}}\|g_{ij}^u\|^2 + \frac{(u_j^i - \overset{*}{u}_j)^2 - (u_j^{i+1} - \overset{*}{u}_j)^2}{\eta_{\text{out}}}\right) \\
&\leq \sum_{j=1}^{T}\frac{1}{2}\left(\eta_{\text{out}}\left(\sum_{i=1}^{N}\|g_{ji}^u\|^2\right) + \frac{(u_j^1 - \overset{*}{u}_j)^2}{\eta_{\text{out}}}\right) \\
&\leq 2UG\kappa\delta^{-2}(\kappa U + \kappa\gamma LW + W)T\sqrt{N} \qquad (5.5.5)
\end{aligned}$$

where the last inequality follows using Lemma 5.5.3 and choice of η_{out}.

Inner Regret: Next we analyze the second Inner Regret term. Before doing so we recommend the reader to re-familiarize with the notations defined in Definition 5.5.2 and Equations 5.5.1,5.5.2,5.5.3. We will also need the following further definitions again for a fixed rollout. Therefore given a dynamical system $AB_{1:T}$, a disturbance sequence $w_{1:T}$, and an open loop sequence $u_{1:T}$ define the notion of surrogate state at time t which is parameterized by a lookback window S and is a function of an input sequence $M_{1:L} \in \mathbb{R}^{d_u \times d_x}$. Intuitively it corresponds to the state achieved by executing the stationary policy $M_{1:L}$ along with $u_{1:T}$ for S time steps, starting at time $t-S$ with a resetted state. This is exactly the computation performed in the GPCLoss definition in Equation 5.4.1. We can use the linear operator ψ defined in Definition 5.5.2 for an alternative and succinct definition as follows.

$$\hat{x}_t(u_{1:T}, M_{1:L}) = \sum_{j=t-S}^{t-1} T_{j\to t}(B_j u_j + w_j) + \psi^M_{t-S\to t}(\mathcal{T}_T M_{1:L}). \tag{5.5.6}$$

Further given a cost function c_t, we can use the above definition to also define a surrogate cost

$$\hat{c}_t(u_{1:T}, M_{1:L}) = c_t\left(\hat{x}_t(u_{1:T}, M_{1:L}), u_t + \sum_{j=1}^{L} M_j w_{t-j}\right) \tag{5.5.7}$$

It can be observed now by the definition of Algorithm 8, the sequence $M^i_{1:T,1:L}$ played by the algorithm is chosen iteratively as follows

$$M^i_{t+1,1:L} = \mathrm{Proj}_{\mathcal{M}_\gamma}\left(M^i_{t,1:L} - \eta_{\text{in}} \nabla_M \hat{c}_t(u^i_{1:T}, M^i_{t,1:L})\right). \tag{5.5.8}$$

To proceed with the proof we will need the following lemma

Lemma 5.5.4. *Consider a linear system $AB_{1:T}$ satisfying Assumption 5.2.2, a bounded disturbance sequence $w_{1:T}$ and a sequence of cost functions $c_{1:T}$ satisfying Assumption 5.2.1. Given any open loop sequence $u_{1:T} \in \mathcal{U}$ and a closed-loop matrix sequence $M_{1:T,1:L} \in \mathcal{M}_\gamma$ generated through the iteration specified in Equation 5.5.8, we have that the following properties hold for all $t \in [T]$*

- *For all $j > t$, $\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L}) = 0$.*
- *For all $j < t$, $\|\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})\| \le \kappa^2 G(U + \gamma LW)LW\delta^{-1}(1-\delta)^{t-j}$.*
- *For all t, $\|\nabla_{M_{1:L}} \hat{c}_t(u_{1:T}, M_{1:L})\| \le GLW(U + \gamma LW)\left(1 + \frac{\kappa^2}{\delta^2}\right)$.*

- *Furthermore, for any $\overset{*}{M}_{1:L} \in \mathcal{M}_\gamma$ and for any t, we have that*

$$\sum_{j=t-S}^{t} \nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{j,1:L} - \overset{*}{M}_{1:L}) \leq \nabla_{M_{t,1:L}} \hat{c}_t(u_{1:T}, M_{t,1:L})(M_{t,1:L} - \overset{*}{M}_{1:L}) + 20\eta_{\text{in}} \log^2(\eta_{\text{in}})\gamma\kappa^4\delta^{-3}\beta G^2 L^3 W^3 (U + \gamma LW)^2$$

We are now ready to analyze the inner regret term. We analyze this term for one particular rollout say i (thereby dropping i from our notation). We get the following series of calculations,

$$\begin{aligned}
&\sum_{t=1}^{T} \Big(\nabla_M c_t(u_{1:T}, M_{1:T,1:L})(M_{1:T,1:L} - \mathcal{T}_T \overset{*}{M}_{1:L}) \Big) \\
&= \sum_{t=1}^{T} \sum_{j=1}^{T} \Big(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{j,1:L} - \overset{*}{M}_{1:L}) \Big) \\
&= \sum_{t=1}^{T} \sum_{j=1}^{t} \Big(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{j,1:L} - \overset{*}{M}_{1:L}) \Big) \\
&\leq \sum_{t=1}^{T} \sum_{j=t-S}^{t} \Big(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{j,1:L} - \overset{*}{M}_{1:L}) \Big) + 2\kappa^2\gamma GLW(U + \gamma LW)\delta^{-2}(1-\delta)^S \\
&\leq \sum_{t=1}^{T} \left(\underbrace{\nabla_{M_{t,1:L}} \hat{c}_t(u_{1:T}, M_{t,1:L})}_{g_t} (M_{t,1:L} - \overset{*}{M}_{1:L}) \right) + 22T\eta_{\text{in}} \log^2(\eta_{\text{in}})\gamma\kappa^4\delta^{-3}\beta G^2 L^3 W^3 (U + \gamma LW)^2,
\end{aligned}$$

where the statements follow via repeated application of Lemma 5.5.4 and the choice of $S = \delta^{-1}\log(\eta_{\text{in}})$. To analyse further once again via a similar argument as in the case of the outer regret regarding projected gradient descent with learning rate η_{in}, we get that,

$$\begin{aligned}
&\sum_{t=1}^{T} \left(\underbrace{\nabla_{M_{t,1:L}} \hat{c}_t(u_{1:T}, M_{1:T,1:L})}_{g_t} (M_{t,1:L} - \overset{*}{M}_{1:L}) \right) \\
&\leq \sum_{t=1}^{T} \left(\frac{\eta_{\text{in}}}{2} \|g_t\|^2 + \frac{\|M_{t,1:L} - \overset{*}{M}_{1:L}\|^2 - \|M_{t+1,1:L} - \overset{*}{M}_{1:L}\|^2}{2\eta_{\text{in}}} \right) \\
&\leq \frac{\eta_{\text{in}} T}{2} \|g_t\|^2 + \frac{\|M_{1,1:L} - \overset{*}{M}_{1:L}\|^2}{2\eta_{\text{in}}}
\end{aligned}$$

Combining the above equations, Equation 5.5.9 and the choice of η_{in}, we get that the inner regret is

bounded as,

$$\sum_{t=1}^{T}\left(\nabla_M c_t(u_{1:T}, M_{1:T,1:L})(M_{1:T,1:L} - \mathcal{T}_T \overset{*}{M}_{1:L})\right) \leq \tilde{O}\left(\sqrt{T\gamma^3\kappa^4\delta^{-3}\beta G^2 L^5 W^3 (U+\gamma LW)^2}\right)$$

Combining the outer and inner regret terms we finish the proof. □

Below, we prove Lemmas 5.5.3 and 5.5.4, thereby finishing the proof of Theorem 5.3.1.

Now, we prove Lemma 5.5.3. Before the proof we establish some other lemmas which will be useful to us.

Lemma 5.5.5. *Given a linear system $AB_{1:T}$ satisfying Assumption 5.2.2, then the transfer matrices defined in Definition 5.5.2 are bounded as follows*

$$\forall j,k \in [T],[j+1,T] \qquad \|T_{j\to k}\| \leq (1-\delta)^{k-j-1}$$

Proof of Lemma 5.5.5. If $k = j+1$ then by definition and Assumption 5.2.2,

$$\|T_{j\to k}\| = \|I\| \leq 1.$$

Otherwise, again by definition and Assumption 5.2.2,

$$\|T_{j\to k}\| \leq \left(\Pi_{t=j+2}^{k}\|A_t\|\right) \leq (1-\delta)^{k-j-1}.$$

□

Lemma 5.5.6. *Given a linear system $AB_{1:T}$ satisfying Assumption 5.2.2, a bounded disturbance sequence $w_{1:T}$ and a cost sequence c_t satisfying Assumption 5.2.1, then for any $\gamma \geq 0$,$\mathcal{U}$, let $u_{1:T} \in \mathcal{U}, M_{1:T,1:L} \in \mathcal{M}_\gamma$ be two sequences, the following bounds hold for x_t, a_t for all t,*

$$\|x_t(u_{1:T}, M_{1:T,1:L})\| \leq \delta^{-1}(\kappa U + \kappa\gamma LW + W),$$

$$\|a_t(u_{1:T}, M_{1:T,1:L})\| \leq U + \gamma LW.$$

Furthermore we have that for all $j,t \in [T]$ we have that

$$\left\|\frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j}\right\| \leq \begin{cases} \kappa(1-\delta)^{t-j-1} & \text{if } j < t \\ 0 & \text{otherwise} \end{cases}$$

$$\left\| \frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j} \right\| = \begin{cases} 1 & \text{if } j = t \\ 0 & \text{otherwise} \end{cases}$$

Furthermore we have that for $j, t \in [T]$ and $r \in [L]$, we have that

$$\left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} \right\| \leq \begin{cases} \kappa W (1-\delta)^{t-j-1} & \text{if } j < t \\ 0 & \text{otherwise} \end{cases}$$

$$\left\| \frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} \right\| \leq \begin{cases} W & \text{if } j = t \\ 0 & \text{otherwise} \end{cases}$$

Proof. From the definition in Equation 5.5.1 it follows that

$$\|a_t(u_{1:T}, M_{1:T,1:L})\| \leq \|u_t\| + \sum_{r=1}^{L} \|M_{t,r}\| \|w_{t-r}\| \leq U + \gamma LW.$$

Also from the definition it follows that

$$\left\| \frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j} \right\| = \|\delta_{jt} I\| = \begin{cases} 1 & \text{if } j = t \\ 0 & \text{otherwise} \end{cases}$$

From the expansion in Equation 5.5.4, we have that

$$\begin{aligned}
&\|x_t(u_{1:T}, M_{1:T,1:L})\| \leq \sum_{j=1}^{t-1} (\|T_{j\to t}(B_j u_j + w_j)\|) + \|\psi_{1\to t}^{M}(M_{1:T,1:L})\| \\
&\leq \sum_{j=1}^{t-1} (\|T_{j\to t}\| \|(B_j u_j + w_j)\|) + \sum_{j=1}^{t-1} \left(\|T_{j\to t}\| \left(\sum_{r=1}^{L} \|M_{j,r}\| \|w_{j-r}\| \right) \right) \quad \text{(Definition 5.5.2 \& } \Delta\text{-inequality)} \\
&\leq (\kappa U + \kappa\gamma LW + W)) \sum_{j=1}^{t-1} (1-\delta)^{t-j-1} \quad \text{(Lemma 5.5.5 and definitions)} \\
&\leq \frac{1}{\delta}(\kappa U + \kappa\gamma LW + W)
\end{aligned}$$

Also from the definition it follows that for $j \geq t$,

$$\frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j} = 0,$$

and if $j < t$, we have that

$$\left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j} \right\| \leq \|T_{j\to t} B_j\| \leq \kappa(1-\delta)^{t-j-1} \qquad \text{(Lemma 5.5.5)}$$

From the definition in Equation 5.5.1 it follows that for any $r \in [L]$

$$\left\| \frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} \right\| = \|\delta_{jt} I \otimes w_{t-r}^\top\| \leq \begin{cases} W & \text{if } j = t \\ 0 & \text{otherwise} \end{cases}$$

From the expansion in Equation 5.5.4, it follows that for any r and $j \geq t$,

$$\frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} = 0,$$

and if $j < t$, we have that

$$\left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} \right\| \leq \|T_{j\to t} B_j (I \otimes w_{j-r}^\top)\| \leq \kappa W (1-\delta)^{t-j-1} \qquad \text{(Lemma 5.5.5)}$$

□

We are now ready to prove Lemma 5.5.3.

Proof of Lemma 5.5.3. Consider the following calculations for all j, t, following from Lemma 5.5.6,

$$\begin{aligned}
&\|\nabla_{u_j} \left(c_t(u_{1:T}, M_{1:T,1:L})\right)\| \\
&\leq G \max(\|x_t(u_{1:T}, M_{1:T,1:L}\|\|a_t(u_{1:T}, M_{1:T,1:L}\|) \left(\left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j} \right\| + \left\| \frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial u_j} \right\| \right) \\
&\leq \begin{cases} G\kappa\delta^{-1}(\kappa U + \kappa\gamma LW + W)(1-\delta)^{t-j-1} & \text{if } j < t \\ G\kappa\delta^{-1}(\kappa U + \kappa\gamma LW + W) & j = t \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

Therefore we have that,

$$\left\| \nabla_{u_j} \left(\sum_{t=1}^{T} c_t(u_{1:T}, M_{1:T,1:L}) \right) \right\| \leq 2G\kappa\delta^{-2}(\kappa U + \kappa\gamma LW + W)$$

□

In this remainder of the subsection we prove Lemma 5.5.4. To this end we will need the following lemma that is the extension of Lemma 5.5.6 to surrogate states.

Lemma 5.5.7. *Given a linear system $AB_{1:T}$ satisfying Assumption 5.2.2, a bounded disturbance sequence $w_{1:T}$ and a cost sequence c_t satisfying Assumption 5.2.1, then for any $\gamma \geq 0$,$\mathcal{U}$, let $u_{1:T} \in \mathcal{U}, M_{1:L} \in \mathcal{M}_\gamma$ be two sequences, then we have that for all $j, t \in [T]$,*

$$\|\hat{x}_t(u_{1:T}, M_{1:L})\| \leq \delta^{-1}(\kappa U + \kappa\gamma LW + W)$$

Furthermore we have that for $t \in [T]$ and $r \in [L]$, we have that

$$\left\| \frac{\partial \hat{x}_t(u_{1:T}, M_{1:L})}{\partial M_r} \right\| \leq \delta^{-1}\kappa W$$

Proof. From the expansion in Equation 5.5.6, we have that

$$\begin{aligned}
\|x_t(u_{1:T}, M_{1:L})\| &\leq \sum_{j=t-S}^{t-1} (\|T_{j\to t}(B_j u_j + w_j)\|) + \|\psi^M_{t-S\to t}(\mathcal{T}_T M_{1:L})\| \\
&\leq \sum_{j=t-S}^{t-1} (\|T_{j\to t}\|\|B_j u_j + w_j\|) + \sum_{j=t-S}^{t-1} \left(\|T_{j\to t}\| \left(\sum_{r=1}^{L} \|M_r\|\|w_{j-r}\| \right) \right) && \text{(Definition 5.5.2 \& } \Delta\text{-inequality)} \\
&\leq (\kappa U + \kappa\gamma LW + W) \sum_{j=t-S}^{t-1} (1-\delta)^{t-j-1} && \text{(Lemma 5.5.5 and Definitions)} \\
&\leq \frac{1}{\delta}(\kappa U + \kappa\gamma LW + W)
\end{aligned}$$

From the expansion in Equation 5.5.6, it follows that

$$\left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_r} \right\| \leq \left\| \sum_{j=t-S}^{t-1} T_{j\to t} B_j I \otimes w_{j-r}^\top \right\| \leq \delta^{-1}\kappa W \qquad \text{(Lemma 5.5.5)}$$

□

We are now ready to prove Lemma 5.5.4.

Proof of Lemma 5.5.4. Since for any $j > t$, by Lemma 5.5.6, we have that

$$\frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,1:L}} = 0, \frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,1:L}} = 0,$$

it immediately follows that for all $j > t$,

$$\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L}) = 0.$$

Furthermore again from Lemma 5.5.6, we have that for all $j < t$ and for all $r \in [L]$,

$$\left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} \right\| \leq \kappa W (1-\delta)^{t-j-1}$$

and further if $j < t$ and for all $r \in [L]$,

$$\frac{\partial a_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} = 0$$

Therefore, since the cost function c_t satisfies the Assumption 5.2.1, using Lemma 5.5.6, we have that for all $j < t$ and for any $r \in [L]$

$$\begin{aligned} \left\| \nabla_{M_{j,r}} c_t(u_{1:T}, M_{1:T,1:L}) \right\| &\leq G \|x_t(u_{1:T}, M_{1:T,1:L})\| \left\| \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,r}} \right\| \\ &\leq G\kappa\delta^{-1} W(\kappa U + \kappa\gamma LW + W)(1-\delta)^{t-j} \end{aligned} \tag{5.5.9}$$

Using Lemma 5.5.7 for the surrogate states and using Assumption 5.2.1, we have that for all t and for all $r \in [L]$,

$$\|\nabla_{M_r} \hat{c}_t(u_{1:T}, M_{1:L})\| \leq 2G\kappa\delta^{-2} W(\kappa U + \kappa\gamma LW + W)$$

Since the gradient is bounded according to the above calculation and the $M_{t,1:L}$ are generated via gradient descent with a learning rate η_{in}, it is immediate that for any $j, k \in [T]$ and for any $r \in [L]$,

$$\|M_{j,r} - M_{k,r}\| \leq \eta_{\text{in}} |j-k| \cdot 2G\kappa\delta^{-2} W(\kappa U + \kappa\gamma LW + W) \tag{5.5.10}$$

Given the above we show that for any execution the surrogate states and the real states are close to

each other. To this end consider the following calculations.

$$
\begin{aligned}
&\|x_t(u_{1:T}, M_{1:T,1:L}) - \hat{x}_t(u_{1:T}, M_{t,1:L})\| \\
\leq &\left\| \sum_{j=1}^{t-1} T_{j\to t}\left(B_j u_j + w_j\right) + \psi^M_{1\to t}(M_{1:T,1:L}) - \sum_{j=t-S}^{t-1} T_{j\to t}\left(B_j u_j + w_j\right) - \psi^M_{t-S\to t}(\mathcal{T}_T M_{t,1:L}) \right\| \\
= &\left\| \sum_{j=1}^{t-S-1} \left(T_{j\to t}\left(B_j u_j + w_j + \sum_{r=1}^{L} M_{j,r} w_{j-r} \right) \right) + \sum_{j=t-S}^{t-1} \left(T_{j\to t}\left(\sum_{r=1}^{L} (M_{j,r} - M_{t,r}) w_{j-r} \right) \right) \right\| \\
&\leq \left(\kappa U + \kappa\gamma LW + W\right)\left(\delta^{-1}(1-\delta)^S + 2\eta_{\text{in}}\kappa\delta^{-2}S^2GLW^2\right) \qquad (5.5.11)
\end{aligned}
$$

Furthermore, note by definitions that

$$
\sum_{j=t-S}^{t-1} \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,1:L}} = \frac{\partial \hat{x}_t(u_{1:T}, M_{t,1:L})}{\partial M_{t,1:L}} \qquad (5.5.12)
$$

Before moving further, consider the following calculations

$$
\begin{aligned}
&\sum_{j=t-S}^{t-1} \left(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})\right) \\
&= \sum_{j=t-S}^{t-1} \left(\frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,1:L}} \nabla_x c_t(x_t(u_{1:T}, M_{1:T,1:L}), a_t(u_{1:T}, M_{1:T,1:L})) \right) \\
&= \sum_{j=t-S}^{t-1} \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,1:L}} \left((\nabla_x c_t(\hat{x}_t(u_{1:T}, M_{t,1:L}), a_t(u_{1:T}, M_{1:T,1:L})) + v) \right)
\end{aligned}
$$

where

$$
\begin{aligned}
\|v\| \triangleq \|\nabla_x c_t(x_t(u_{1:T}, M_{1:T,1:L}), a_t(u_{1:T}, M_{1:T,1:L})) - \nabla_x c_t(\hat{x}_t(u_{1:T}, M_{t,1:L}), a_t(u_{1:T}, M_{1:T,1:L}))\| \\
\leq \beta(\kappa U + \kappa\gamma LW + W)\left(\delta^{-1}(1-\delta)^S + 2\eta_{\text{in}}\kappa\delta^{-2}S^2GLW^2\right) \qquad (5.5.13)
\end{aligned}
$$

using Equation 5.5.11 and the β-smoothness of c_t via Assumption 5.2.1. Using Equation 5.5.12 and Lemma 5.5.6 we now get that

$$
\begin{aligned}
&\sum_{j=t-S}^{t-1} \left(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})\right) \\
&= \left(\sum_{j=t-S}^{t-1} \frac{\partial x_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{j,1:L}} \right) \left(\nabla_x c_t(\hat{x}_t(u_{1:T}, M_{t,1:L}), a_t(u_{1:T}, M_{1:T,1:L})) + v \right) \\
&= \frac{\partial \hat{x}_t(u_{1:T}, M_{1:T,1:L})}{\partial M_{t,1:L}} \left(\nabla_x c_t(\hat{x}_t(u_{1:T}, M_{t,1:L}), a_t(u_{1:T}, M_{1:T,1:L})) \right) + v' \qquad (5.5.14)
\end{aligned}
$$

where v' is a vector whose norm using Equation 5.5.13 and Lemma 5.5.7 can be bounded as follows

$$\beta\delta^{-1}LW(\kappa U + \kappa\gamma LW + W)\left(\delta^{-1}(1-\delta)^S + 2\eta_{\text{in}}\kappa\delta^{-2}S^2GLW^2\right). \tag{5.5.15}$$

Now, consider the following computation which follows from Equation 5.5.14 and using the defintiions for the $j = t$ case,

$$\sum_{j=t-S}^{t}\left(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})\right) = \nabla_{M_{t,1:L}}\hat{c}_t(u_{1:T}, M_{t,1:L}) + v'. \tag{5.5.16}$$

We can now perform the calculation to relate the gradient inner products for surrogate cost to those of real cost.

$$\begin{aligned}
&\sum_{j=t-S}^{t}\left(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{j,1:L} - \overset{*}{M}_{1:L})\right)\\
&= \sum_{j=t-S}^{t}\left(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{t,1:L} - \overset{*}{M}_{1:L}) + \nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{j,1:L} - M_{t,1:L})\right)\\
&\leq \sum_{j=t-S}^{t}\left(\nabla_{M_{j,1:L}} c_t(u_{1:T}, M_{1:T,1:L})(M_{t,1:L} - \overset{*}{M}_{1:L})\right) + \eta_{\text{in}} 2G^2LS^2\kappa^2\delta^{-3}W^2(\kappa U + \kappa\gamma LW + W)^2\\
&\leq \nabla_{M_{t,1:L}}\hat{c}_t(u_{1:T}, M_{t,1:L})(M_{t,1:L} - \overset{*}{M}_{1:L}) +\\
&\qquad \beta\delta^{-1}\gamma L^2W(\kappa U + \kappa\gamma LW + W)^2\left(\delta^{-1}(1-\delta)^S + 4\eta_{\text{in}}\kappa^2\delta^{-2}S^2G^2LW^2\right)\\
&\leq \nabla_{M_{t,1:L}}\hat{c}_t(u_{1:T}, M_{t,1:L})(M_{t,1:L} - \overset{*}{M}_{1:L}) + 5\eta_{\text{in}}\log^2(\eta_{\text{in}})\gamma\kappa^2\delta^{-3}\beta G^2L^3W^3(\kappa U + \kappa\gamma LW + W)^2
\end{aligned}$$

where the first inequality follows from applying Equations 5.5.9, 5.5.10 and Lemma 5.5.6, the second last inequality follows from Equations 5.5.15 and 5.5.16 and the last inequality follows from the choice of the parameter $S = \delta^{-1}\log(\eta_{\text{in}})$. This finishes the proof. □

5.6 Experiments

We demonstrate the efficacy of the proposed approach on two sets of experiments: the theory-aligned one performs basic checks on linear dynamical systems; the subsequent set demonstrates the benefit on highly non-linear systems distilled from practical applications. In the following we provide a detailed description of the setup and the results are presented in Figure 5.1.

5.6.1 Experimental setup

We briefly review the methods that we compare to: The **ILQG** agent obtains a closed loop policy via the Iterative Linear Quadratic Gaussian algorithm [TL05], proposed originally to handle Gaussian noise while planning on non-linear systems, on the simulator dynamics, and then executes the policy thus obtained. This approach does not *learn* from multiple rollouts and, if the dynamics are fixed, provides a constant (across rollouts) baseline.

The Iterative Learning Control (**ILC**) agent [AQN06] *learns* from past trajectories to refine its actions on the next real-world rollout. We provide precise details in a following subsection (Section 5.6.4). Finally, the **IGPC** agent adapts Algorithm 7 by replacing the policy update step (Line 5) with a LQR step on locally linearized dynamics.

In all our experiments, the metric we compare is the number of real-world rollouts required to achieve a certain loss value on the real dynamics.

5.6.2 Linear control

This section considers a discrete-time **Double Integrator** (detailed below), a basic kinematics model well studied in control theory. This linear system (described below) is subject to a variety of perturbations that vary either within or across episodes,

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

We pick three instructive perturbation models: First, as a sanity check, we consider constant offsets. While both ILC and IGPC adapt to this change, IGPC is quicker in doing so as evident by the cost on the first rollout itself. In the second, we treat constant offsets that gradually increase in magnitude from zero with rollouts/episodes. While gradual inter-episodic changes are well suited to ILC, IGPC still offers consistently better performance. The final scenario considers time-varying sinusoidal perturbations subject to rollout-varying phase shifts. In contrast to the former setups, such conditions make intra-episodic learning crucial for good performance. Indeed, IGPC outperforms alternatives here by a margin, reflecting the benefit of rollout-adaptive feedback policy in the regret bound.

5.6.3 Non-linear control with approximate models

Here, we consider the task of controlling non-linear systems whose real-world characteristics are only partially known. In the cases presented below, the proposed algorithm **IGPC** either converges to the optimal cost with fewer rollouts (for Quadrotor), or, even disregarding speed of convergence, offers a better terminal solution quality (for Reacher). These effects are generally more pronounced in situations where the model mismatch is severe.

Concretely, consider the following setup: the agent is scored on the cost incurred on a handful of sequentially executed real-world rollouts on a dynamical system $g(x, u)$; all the while, the agent has access to an inaccurate simulator $f(x, u) \neq g(x, u)$. In particular, while limited to simply observing its trajectories in the real world g, the agent is permitted to compute the function value and Jacobian of the simulator $f(x, y)$ along arbitrary state-action pairs. The disturbances here are thus the difference between g and f along the state-action pairs visited along any given real world rollout. Here, we also consider a statistically-omnipotent *infeasible agent* **ILQR (oracle)** that executes the Iterative Linear Quadratic Regulator algorithm [LT04] directly via Jacobians of the real world dynamics g (a cheat), indicating a lower bound on the best possible cost.

Quadrotor with Wind The simulator models an underactuated planar quadrotor (6 dimensional state, 2 dimensional control) attempting to fly to $(1, 1)$ from the origin. The real-world dynamics differ from the simulator in the presence of a dispersive force field $(x\hat{\mathbf{i}} + y\hat{\mathbf{j}})$, to accomodate wind. The cost is measured as the distance sqaured from the origin along with a quadratic penalty on the actions.

Reacher with Impulse The simulator dynamics model a 2-DOF arm (6 dimensional state, 2 dimensional control) attempting to place its end-effector at a pre-specified goal. The true dynamics g differs from the simulator in the application of periodic impulses to the center of mass of the arm links. The cost involves a quadratic penalty on the controls and the distance of the end effector from the goal.

In both scenarios, JAX-based [BFH+18] differentiable implementations of the underlying dynamics were adapted from [GHS+21].

5.6.4 Details of ILQR, ILC and IGPC algorithms

To succinctly state the algorithms define the following policy which takes as arguments a nominal trajectory $\mathring{x}_{1:T} \in \mathbb{R}^{d_x}, \mathring{u}_{1:T} \in \mathbb{R}^{d_u}$, open-loop gain seqeunce $k_{1:T}$ and closed-loop gain sequence $K_{1:T}$

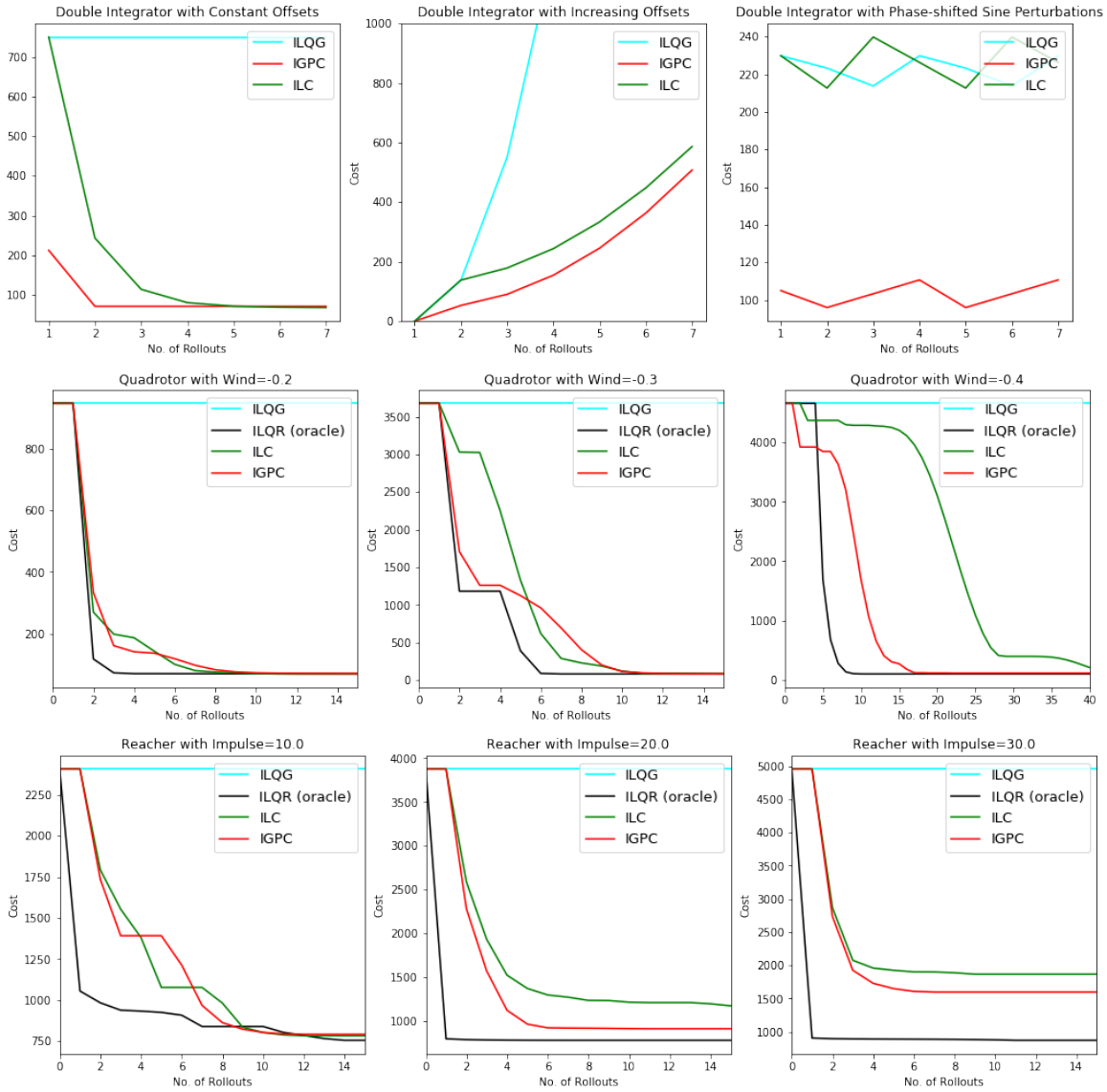


Figure 5.1: On top is a linear system, Double Integrator, setup subject to: (L) constant offset, (M) offset that increments with rollout count, (R) phase-shifted sinusoidal perturbations. The middle section displays results on the quadrotor environment for varying magnitudes of wind. Bottom figure captures performance on the reacher environment with varying magnitudes of periodic impulses. **ILQR (oracle)** is an infeasible agent with access to Jacobians on the real world.

and a parameter α. The policy defined as $\pi(\alpha, x_{1:T}, k_{1:T}, K_{1:T})$, in the sequel executes the following *standard* rollout on a dynamical system $f_{1:T}$.

$$a_t = \mathring{u}_t + \alpha k_t + K(x_{t-1} - \mathring{x}_{t-1})$$

$$x_{t+1} = f_t(x_t, a_t)$$

Before stating the algorithm we also need the following quadratic approximation of the cost function c around pivots x_0, u_0

$$Q(c, x_0, u_0)(x, u) \triangleq \nabla c_x(x_0, u_0)(x - x_0) + \nabla c_u(x_0, u_0)(u - u_0) + \frac{1}{2}([x, u] - [x_0, u_0])^\top \nabla^2 c(x, u)([x, u] - [x_0, u_0]) \quad (5.6.1)$$

Algorithm 10 now presents a combined layout for ILQG,ILC and IGPC.

Algorithm 10 Iterative Planning Algorithm

1: Inputs: $g_{1:T}$ Real Dynamical Systems, $f_{1:T}$ Simulator.
2: Initialize starting sequence of actions $u^0_{1:T}$
3: Initialize sequence of open loop $k^0_{1:T} = 0$ and closed loop gains $K^0_{1:T} = 0$.
4: **for** $i = 1 \dots N$ **do**
5: **Rollout the Policy**:

- **ILQG:** Standard Rollout on $f_{1:T}$.

$$x^i_{1:T}, u^i_{1:T} = \text{Rollout}(f_{1:T}, \pi(\alpha, x^{i-1}_{1:T}, u^{i-1}_{1:T}, k^{i-1}_{1:T}, K^{i-1}_{1:T}))$$

- **ILC:** Standard Rollout on $g_{1:T}$.

$$x^i_{1:T}, u^i_{1:T} = \text{Rollout}(g_{1:T}, \pi(\alpha, x^{i-1}_{1:T}, u^{i-1}_{1:T}, k^{i-1}_{1:T}, K^{i-1}_{1:T}))$$

.

- **IGPC:** GPCRollout on $g_{1:T}$,

$$x^i_{1:T}, u^i_{1:T} = \text{GPCRollout}(g_{1:T}, \pi(\alpha, x^{i-1}_{1:T}, u^{i-1}_{1:T}, k^{i-1}_{1:T}, K^{i-1}_{1:T}))$$

6: **Update**: Obtain $k^i_{1:T} \in \mathbb{R}^{d_u}, K^i_{1:T} \in \mathbb{R}^{d_u \times d_x}$ as the optimal non-stationary affine policy to the following LQG problem.

$$\min \mathbb{E}_z \left[\sum_{t=1}^{T} Q(c_t, x^i_t, u^i_t)(x_t, u_t) \right]$$
$$\text{subject to} \qquad x_{t+1} - x^i_{t+1} = \frac{\partial f_t(x^i_t, u^i_t)}{\partial x^i_t}(x_t - x^i_t) + \frac{\partial f_t(x^i_t, u^i_t)}{\partial u^i_t}(u_t - u^i_t) + z_t$$

where z_t are independent Gaussians of any non-zero variance.
7: **end for**

5.6.5 Hyperparameter choices for experiments

ILQG, ILC, IGPC in particular share one hyperparameter α which corresponds essentially to a step size towards the updated policy. As is common in implementations, this hyperparameter is adjusted online during the run of the algorithm using a simple retracting line search from a certain upper bound α^+. We optimize over choices for α^+ for ILC and report the best performance obtained as

baseline. For IGPC, we use the same α^+ as obtained for ILC and the same line search for strategy for selecting α. We include the rollouts needed for line search in the rollout cost of the algorithm. Further, IGPC introduces certain other hyperparameters, L the window, S the lookback, and η_{in}, the inner learning rate. We chose $L, S = 3$ arbitrarily for our experiments and tuned η_{in} per experiment. Overall we observed that for every experiment, the selection of η was robust in terms of performance.

Part III

Beyond Scalar Rewards

Chapter 6

Boosting for Online Convex Optimization

This chapter considers the decision-making framework of online convex optimization with a very large number of experts. This setting is ubiquitous in contextual and reinforcement learning problems, where the size of the policy class renders enumeration and search within the policy class infeasible.

In this context, we consider generalizing the methodology of online boosting. A weak learning algorithm is a mechanism that guarantees multiplicatively approximate regret against a base class of experts. Given access to such weak learning algorithms, the main results in this chapeter is an efficient boosting algorithm that guarantees near-optimal regret against the convex hull of the base class. We consider both full and partial (a.k.a. bandit) information feedback models. We also give an analogous efficient boosting algorithm for the i.i.d. statistical setting.

The results simultaneously generalize online boosting and gradient boosting guarantees to contextual learning model, online convex optimization and bandit linear optimization settings.

6.1 Introduction

In the classical problem of prediction from expert advice [CBFH$^+$97], a learner iteratively makes decisions and receives loss according to an arbitrarily chosen loss function. Learning in terms of guaranteeing an absolute bound on the loss incurred would be a hopeless task, but the learner is assisted by a pool of experts (or hypotheses). The goal of the learner is thus to minimize regret, or the difference of total loss to that of the best expert in hindsight.

For this canonical problem, numerous methods have been devised and refined, some notably based on the multiplicative weights algorithm [LW94, AHK12]. It is well established that the regret can be bounded by $O(\sqrt{T \log |\mathcal{H}|})$, where $|\mathcal{H}|$ is the number of experts, and that this is tight. However, in many problems of interest, the class of experts is too large to efficiently manipulate. This is particularly evident in contextual learning, where the experts are *policies* – functions mapping contexts to action. In such instances, even if a regret bound of $O(\sqrt{T \log |\mathcal{H}|})$ is meaningful, the algorithms achieving this bound are computationally inefficient; their running time is linear in $|\mathcal{H}|$.

One approach to deal with this computational difficulty is to grant the leaner a best-in-hindsight (or ERM) oracle. This has been explored in both the stochastic bandit [LZ08, AHK$^+$14, DHK$^+$11] and (transductive) online settings [RS16, SLKS16, SKS16, HK16].

In this chapter, we consider a different approach towards addressing computational intractability, motivated by the observation that it is often possible to design simple *rules-of-thumb* [VJ01] that perform slightly better than random guesses. We propose that the learner has access to a "weak learner" - an computationally cheap mechanism capable of guaranteeing multiplicatively approximate regret against a base hypotheses class. Such considerations arise in the theory of boosting [SF12a], which was originally devised in the context of binary classification in the statistical/i.i.d. setting. A weak learner, defined as one which is capable of better-than-random classification, is used by a boosting algorithm to create an accurate classifier. By analogy, the regret-based notion of weak learner we define here is a natural extension of the concept of weak learning to online convex optimization.

We design efficient algorithms that when provided weak learners compete with the convex hull of the base hypotheses class with near-optimal regret. We further consider different information feedback models: both full gradient feedback, as well as linear bandit (function value) feedback, and derive efficient sublinear regret algorithms for these settings.

6.1.1 Setting and contributions

The setting of online convex optimization (OCO) generalizes the problem of prediction from expert advice to a general convex decision set $\mathcal{K} \subseteq \mathbb{R}^d$, and adversarially chosen convex loss functions $f_t : \mathbb{R}^d \mapsto \mathbb{R}$. We consider a hypothesis class $\mathcal{H} \subseteq \mathcal{C} \mapsto \mathcal{K}$, where each hypothesis $h \in \mathcal{H}$ maps a context (or feature vector) in the context set $c_t \in \mathcal{C}$ to an action $h(c_t) \in \mathcal{K}$.

Definition 6.1.1. *An online learning algorithm $\mathcal{W}$ is a γ-**weak OCO learner (WOCL)** for a class $\mathcal{H}$ and edge $\gamma \in (0,1)$, if for any sequence of contexts chosen from $\mathcal{C}$ and unit*[1] linear *loss*

[1] $f(x)$ is a unit loss function if $\max_{x \in \mathcal{K}} f(x) - \min_{x \in K} f(x) \leq 1$. This scaling is w.l.o.g. and all results hold with a different constant otherwise.

functions $f_1, ..., f_T$*:*

$$\sum_{t=1}^{T} f_t(\mathcal{W}(c_t)) \leq \gamma \cdot \min_{h \in \mathcal{H}} \sum_{t=1}^{T} f_t(h(c_t)) + (1-\gamma) \sum_{t=1}^{T} f_t^{\mu} + Regret_T(\mathcal{W}),$$

where $f^{\mu} = \int_{\mathbf{x} \in \mathcal{K}} f(\mathbf{x}) d\mu$ *is the average value of* f *under the uniform distribution* μ*.*

In classical boosting, a weak learner is an algorithm that offers an edge over a random guess. Since f_t^{μ} is precisely the loss associated with a uniform random guess, the above definition is a generalization of the classic notion. Also, such a definition is invariant under constant offsets to the function value.

It is convenient to henceforth assume that the convex decision set is centered, that is $\int_{\mathbf{x} \in K} \mathbf{x} d\mu = 0$. If so, note that for any linear function $f : \mathcal{R}^d \to \mathcal{R}$, $f_t^{\mu} = f(\int_{\mathbf{x}} \mathbf{x} d\mu) = 0$. With this adjustment, we can rephrase the γ-WOCL learning promise as:

$$\sum_{t=1}^{T} f_t(\mathcal{W}(c_t)) \leq \gamma \cdot \min_{h \in \mathcal{H}} \sum_{t=1}^{T} f_t(h(c_t)) + \text{Regret}_T(\mathcal{W}). \tag{6.1.1}$$

Let $\text{CH}(\mathcal{H})$ be the convex hull of the hypothesis class. Our goal is to design an algorithm $\mathcal{A}$ which minimizes regret vs. the best hypothesis in the convex hull of the base class.

$$\text{Regret} = \sum_{t=1}^{T} f_t(\mathcal{A}(c_t)) - \min_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t)).$$

In this model, our main result is an efficient algorithm, whose guarantee is given by:

Theorem 6.1.2 (Main)**.** *Algorithm 11 with parameters* γ, N*, maintains* N *copies of* γ*-WOCL, and generates a sequence actions* $\mathbf{x}_1, ..., \mathbf{x}_T$*, such that*

$$Regret = O\left(\frac{T}{\gamma\sqrt{N}} + \frac{1}{\gamma} Regret_T(\mathcal{W})\right).$$

The formal theorem which we state and prove below has explicit constants that depend on the diameter of $\mathcal{K}$ (as opposed to $|\mathcal{H}|$) and the Lipschitz constants of the loss functions. These are hidden for brevity. Notably, Algorithm 11 achieves the following goals:

1. Its running time does **not** depend on $|\mathcal{H}|$, but rather only on the other natural parameters of the problem, notably the parameter N.

2. The regret it guarantees holds with respect to a stronger benchmark than the best single

hypothesis in the base class. It competes with the best convex combination of hypotheses from the base class.

3. Finally, the regret guarantee is not multiplicatively approximate anymore; it is directly with respect to the best convex combination without the γ factor.

6.1.2 Related work

Boosting and ensemble methods are fundamental in all of machine learning and data science, see e.g. [Ols01] for a retrospective viewpoint and [SF12b] for a comprehensive text. Originally boosting was studied for statistical learning and the realizable setting. For a framework that encompasses the different boosting methods see [BCHM20].

More recently boosting was extended to real-valued prediction via the theory of gradient boosting [Fri02], and to the online setting [LSRB09, CLL12, CLL14, BKL15, BHKL15, ABHL19, JGT17, JT18, BH20, BHS21, AS17, HSZ17a]. In each of these works, a certain component prevents the result from applying to the full OCO setting. Either the weak learner has a parameter $\gamma = 1$, the convex decision sets are restricted, or the loss functions are restricted.

Our work is the first to generalize online boosting to real-valued prediction in the full setting of online convex optimization, that allows for arbitrary convex decision sets and arbitrary approximation factor γ. This requires the use of a new extension operator, defined henceforth.

For extensive background on online learning and online convex optimization see [Haz19, SS$^+$12].

The contextual experts and bandits problems have been proposed in [LZ08] as a decision making framework with large number of policies. In the online setting, several works study the problem with emphasis on efficient algorithms given access to an ERM oracle [RS16, SLKS16, SKS16, RS16].

Another related line of work in online learning studies the possibility of α-regret minimization given access to approximation oracles with an approximate weak learner [KKL09, HHLL18]. This weaker notion of α-regret only generates approximately optimal decisions. In contrast, our result gives a stronger guarantee of standard regret, and the regret is compared to a stronger benchmark of convex combination of experts.

6.2 Improper learning and the extension operator

6.2.1 Preliminaries

For a convex set $\mathcal{K}$ and scalar $c > 0$ we denote by $c\mathcal{K}$ the set of all scaled points, i.e.

$$c\mathcal{K} = \{c\mathbf{x} \text{ s.t. } \mathbf{x} \in \mathcal{K}\}.$$

The algorithmic primitives introduced next use a smoothing operator. Let f be G-Lipschitz continuous. The smoothing operator may be defined via inf-convolution or Moreau-Yoshida regularization:

$$M_\delta[f] = \inf_{y \in \mathbb{R}^d} \left\{ f(y) + \frac{1}{2\delta}\|x - y\|^2 \right\}.$$

The following lemma elucidates well known properties of such an operator (see [Bec17]).

Lemma 6.2.1. *The smoothed function $\hat{f}_\delta = M_\delta[f]$ satisfies:*

1. *$\hat{f}_\delta$ is $\frac{1}{\delta}$-smooth, and G-Lispchitz.*
2. *$|\hat{f}_\delta(\mathbf{x}) - f(\mathbf{x})| \leq \frac{\delta G^2}{2}$ for all $\mathbf{x} \in \mathbb{R}^d$.*

6.2.2 The extension operator

The main difficulty we face is that the weak learner only has a γ-approximate regret guarantee. To cope with this, the algorithm we describe henceforth scales the predictions returned by the weak learner by a factor of $\frac{1}{\gamma}$. This algorithm, a rescaled variant of the Frank-Wolfe method, guarantees competitiveness with the convex hull of the base class, if we could somehow play actions in $\frac{1}{\gamma}\mathcal{K}$, instead of $\mathcal{K}$. Since these actions do not belong to the decision set, they are infeasible.

This presents a significant challenge. Ideally, we would like that for every action in $\gamma^{-1}\mathcal{K}$, one could find an action in $\mathcal{K}$ of comparable (or lesser) loss (for all loss functions). It can be seen that some natural families of functions, i.e. linear functions, do not admit such an operation. For linear loss functions, for example, extrema always occur on the boundary of the set.

To remedy this, we modify the loss function being fed to the Frank-Wolfe algorithm in the first place. We call this modification an *extension* of the loss function outside $\mathcal{K}$. Define $\mathbf{P}(\mathbf{x}, \mathcal{K})$ the

Euclidean distance to $\mathcal{K}$, and the Euclidean projection $\Pi_{\mathcal{K}}$ as

$$\mathbf{P}(\mathbf{x}, K) = \min_{\mathbf{y}\in\mathcal{K}} \|\mathbf{y} - \mathbf{x}\|,$$

$$\Pi_{\mathcal{K}}(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{y}\in\mathcal{K}} \|\mathbf{y} - \mathbf{x}\|$$

Definition 6.2.2 ($(\mathcal{K}, \delta, \kappa)$-extension)**.** *The extension operator over $\mathcal{K} \subseteq \mathbb{R}^d$ is defined as:*

$$X_{\mathcal{K},\delta,\kappa}[f](x) = M_\delta[f(x) + \kappa \cdot \mathbf{P}(\mathbf{x}, \mathcal{K})].$$

We henceforth denote by G an upper bound on the norm of the gradients of the loss functions over $\mathcal{K}$, and set $\kappa = G = \max_t \max_{\mathbf{x}\in\mathbb{R}^d} \|\nabla f_t(\mathbf{x})\|$. The important take-away from these operators is the following lemma, whose importance is crucial in the OCO boosting algorithm 11. The extended loss functions are smooth, agree with the original loss on $\mathcal{K}$, and possess the property that projection of any action in $\mathcal{R}^d$ onto $\mathcal{K}$ does not increment the extended loss value. In short, it permits projections of infeasible points that are obtained from the weak learners to the feasible domain without an increase in cost.

Lemma 6.2.3. *The $(\mathcal{K}, \delta, \kappa)$-extension of a G-lipschitz function $\hat{f} = X_{K,\delta,\kappa}[f]$ satisfies the following:*

1. *For every point $\mathbf{x} \in \mathcal{K}$, we have $|\hat{f}(\mathbf{x}) - f(\mathbf{x})| \leq \frac{\delta G^2}{2}$.*
2. *The projection of a point $\mathbf{x}$ onto $\mathcal{K}$ does not increase the $(\mathcal{K}, \kappa, \delta)$-extended function value by more than $G^2\delta$.*

$$\hat{f}\left(\Pi_{\mathcal{K}}(\mathbf{x})\right) \leq \hat{f}(\mathbf{x}) + G^2\delta.$$

Proof. Since $\mathbf{P}(\mathbf{x}, \mathcal{K}) = 0$ for all $x \in \mathcal{K}$, the first claim follows immediately from the properties of the smoothing operator as stated in Lemma 6.2.1. Denote $\mathbf{x}_\pi = \Pi_{\mathcal{K}}(\mathbf{x})$.

$$\begin{aligned}
\hat{f}(\mathbf{x}_\pi) - \hat{f}(\mathbf{x}) &\leq f(\mathbf{x}_\pi) - f(\mathbf{x}) - \kappa \cdot \mathbf{P}(\mathbf{x}, \mathcal{K}) + G^2\delta \\
&= f(\mathbf{x}_\pi) - f(\mathbf{x}) - \kappa\|\mathbf{x} - \mathbf{x}_\pi\| + G^2\delta \\
&\leq G\|\mathbf{x} - \mathbf{x}_\pi\| - \kappa\|\mathbf{x} - \mathbf{x}_\pi\| + G^2\delta = G^2\delta
\end{aligned}$$

Here the first inequality follows from Lemma 6.2.1.2, and the second follows from the lipschitz assumptions on f. The last equality results from the choice $\kappa = G$. □

6.3 Algorithm and main theorems

Algorithm 11 efficiently converts a weak online learning algorithm into an OCO algorithm with vanishing regret in a black-box manner. The idea is to apply the weak learning algorithm on linear functions that are gradients of the loss. The algorithm then recursively applies another weak learner on the gradients of the residual loss, and so forth.

Algorithm 11 BoOCO = Boosting Online Convex Opt.

Input: N copies of the γ-WOCL $\mathcal{W}^1, \mathcal{W}^2, \ldots, \mathcal{W}^N$, parameters $(\eta_1, \ldots \eta_N)$, δ, $\kappa = G$.
for $t = 1$ **to** T **do**
  Receive context c_t.
  Choose $\mathbf{x}_t^0 \in \mathcal{K}$ arbitrarily.
  for $i = 1$ **to** N **do**
    Define $\mathbf{x}_t^i = (1 - \eta_i)\mathbf{x}_t^{i-1} + \eta_i \frac{1}{\gamma} \mathcal{W}^i(c_t)$.
  end for
  Predict $\mathbf{x}_t = \Pi_{\mathcal{K}}(\mathbf{x}_t^N)$ and suffer loss $f_t(\mathbf{x}_t)$.
  Obtain loss function f_t, create $\hat{f}_t = X_{\mathcal{K},\delta,\kappa}[f_t]$.
  for $i = 1$ **to** N **do**
    Pass to $\mathcal{W}^i$ the linear loss function f_t^i.

$$f_t^i(\mathbf{x}) = \nabla \hat{f}_t(\mathbf{x}_t^{i-1}) \cdot \mathbf{x}.$$

  end for
end for

The main performance guarantee we prove is summarized in the following theorem.

Theorem 6.3.1 (Main)**.** *The actions $\mathbf{x}_t$ generated by Algorithm 11 with $\delta = \sqrt{\frac{D^2}{\gamma N}}, \eta_i = \min\{\frac{2}{i}, 1\}$ satisfy*

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) \; - \min_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t)) \leq \frac{4GDT}{\gamma\sqrt{N}} + \frac{2GD}{\gamma} Regret_T(\mathcal{W}).$$

6.4 Analysis

Define $\hat{f}_t = X[f_t] = M_\delta[f_t + G \cdot \mathbf{P}(\mathbf{x}, \mathcal{K})]$. We apply the setting of $\kappa = G$, as required by Lemma 6.2.3, and by Lemma 6.2.1, $\hat{f}_t$ is $\frac{1}{\delta}$-smooth.

Let $h^* = \text{argmin}_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t))$ be the best hypothesis in the convex hull in hindsight. We define $\mathbf{x}_t^* = h^*(c_t)$ as the decisions of this hypothesis.

The main crux of the proof is given by the following lemma.

Lemma 6.4.1. *Suppose $\hat{f}_t$'s are β-smooth, and $\hat{G}$ lipschitz. Then,*

$$\sum_{t=1}^{T} \hat{f}_t(\mathbf{x}_t^N) \ - \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}_t^\star) \leq \frac{2\beta D^2 T}{\gamma^2 N} + \frac{\hat{G}D}{\gamma} Regret_T(\mathcal{W}).$$

Proof. Define for all $i = 0, 1, 2, \ldots, N$,

$$\Delta_i = \sum_{t=1}^{T} \left(\hat{f}_t(\mathbf{x}_t^i) - \hat{f}_t(\mathbf{x}_t^\star) \right).$$

Recall that $\hat{f}_t$ is β smooth by our assumption. Therefore:

$$\begin{aligned}
\Delta_i =& \sum_{t=1}^{T} \left[\hat{f}_t(\mathbf{x}_t^{i-1} + \eta_i(\frac{1}{\gamma}\mathcal{W}^i(c_t) - \mathbf{x}_t^{i-1})) - \hat{f}_t(\mathbf{x}_t^\star) \right] \\
\leq& \sum_{t=1}^{T} \left[\hat{f}_t(\mathbf{x}_t^{i-1}) - \hat{f}_t(\mathbf{x}_t^\star) + \eta_i \nabla \hat{f}_t(\mathbf{x}_t^{i-1}) \cdot (\frac{1}{\gamma}\mathcal{W}^i(c_t) - \mathbf{x}_t^{i-1}) + \frac{\eta_i^2 \beta}{2} \| \frac{1}{\gamma}\mathcal{W}^i(c_t) - \mathbf{x}_t^{i-1} \|^2 \right]
\end{aligned}$$

By using the definition and linearity of f_t^i, we have

$$\begin{aligned}
\Delta_i \leq& \sum_{t=1}^{T} \left[\hat{f}_t(\mathbf{x}_t^{i-1}) - \hat{f}_t(\mathbf{x}_t^\star) + \eta_i (f_t^i(\frac{1}{\gamma}\mathcal{W}^i(c_t)) - f_t^i(\mathbf{x}_t^{i-1})) + \frac{\eta_i^2 \beta D^2}{2\gamma^2} \right] \\
=& \Delta_{i-1} + \sum_{t=1}^{T} \eta_i (\frac{1}{\gamma} f_t^i(\mathcal{W}^i(c_t)) - f_t^i(\mathbf{x}_t^{i-1})) + \sum_{t=1}^{T} \frac{\eta_i^2 \beta D^2}{2\gamma^2}
\end{aligned}$$

Now, note the following equivalent restatement of the WOCL guarantee, which again utilizes linearity of f_t^i to conclude: linear loss on a convex combination of a set is equal to the same convex combination of the linear loss applied to individual elements.

$$\begin{aligned}
\frac{1}{\gamma} \sum_{t=1}^{T} f_t^i(\mathcal{W}^i(c_t)) \leq& \min_{h \in \mathcal{H}} \sum_{t=1}^{T} f_t^i(h(c_t)) + \frac{\hat{G}D\text{Regret}_T(\mathcal{W})}{\gamma} \\
=& \min_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t^i(h(c_t)) + \frac{\hat{G}D\text{Regret}_T(\mathcal{W})}{\gamma}
\end{aligned}$$

Using the above and that $h^* \in CH(\mathcal{H})$, we have

$$\begin{aligned}
\Delta_i \leq& \Delta_{i-1} + \sum_{t=1}^{T} [\eta_i \nabla \hat{f}_t(\mathbf{x}_t^{i-1}) \cdot (\mathbf{x}_t^\star - \mathbf{x}_t^{i-1}) + \frac{\eta_i^2 \beta D^2}{2\gamma^2}] + \eta_i \frac{\hat{G}D}{\gamma} \text{Regret}_T(\mathcal{W}) \\
\leq& \Delta_{i-1}(1 - \eta_i) + \frac{\eta_i^2 \beta D^2 T}{2\gamma^2} + \eta_i R_T
\end{aligned}$$

where the last inequality uses the convexity of $\hat{f}_t$ and $R_T = \frac{\hat{G}D}{\gamma}\text{Regret}_T(\mathcal{W})$. We thus have the

recurrence

$$\Delta_i \leq \Delta_{i-1}(1-\eta_i) + \eta_i^2 \frac{\beta D^2 T}{2\gamma^2} + \eta_i R_T.$$

Denoting $\hat{\Delta}_i = \Delta_i - R_T$, we are left with

$$\hat{\Delta}_i \leq \hat{\Delta}_{i-1}(1-\eta_i) + \eta_i^2 \frac{\beta D^2 T}{2\gamma^2}.$$

Lemma 6.4.2. *Let $\{h_t\}$ be a sequence that satisfies the recurence*

$$h_{t+1} \leq h_t(1-\eta_t) + \eta_t^2 c.$$

Then taking $\eta_t = \min\{1, \frac{2}{t}\}$ implies

$$h_t \leq \frac{4c}{t}.$$

This is a recursive relation that can be simplified by applying Lemma 6.4.2, taken from [Haz19]. . Specifically, we obtain that $\hat{\Delta}_N \leq \frac{2\beta D^2 T}{\gamma^2 N}$. □

We are ready to prove the main guarantee of Algorithm 11.

Proof of Theorem 6.3.1. Using both parts of Lemma 6.2.3 in succession, we have

$$\begin{aligned}\sum_{t=1}^{T} f_t(\mathbf{x}_t) \ - \sum_{t=1}^{T} f_t(\mathbf{x}_t^\star) &\leq \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}_t) \ - \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}_t^\star) + \delta G^2 T \\ &\leq \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}_t^N) \ - \sum_{t=1}^{T} \hat{f}_t(\mathbf{x}_t^\star) + 2\delta G^2 T\end{aligned}$$

Next, recall by Lemma 6.2.1, that $\hat{f}_t$ is $\frac{1}{\delta}$-smooth. By applying Lemma 6.4.1, and optimizing δ, we have

$$\begin{aligned}\sum_{t=1}^{T} f_t(\mathbf{x}_t) \ - \sum_{t=1}^{T} f_t(\mathbf{x}_t^\star) &\leq 2\delta G^2 T + \frac{2D^2 T}{\delta \gamma^2 N} + \frac{\hat{G} D}{\gamma} \mathrm{Regret}_T(\mathcal{W}) \\ &= \frac{4GDT}{\gamma\sqrt{N}} + \frac{\hat{G} D}{\gamma} \mathrm{Regret}_T(\mathcal{W})\end{aligned}$$

Lemma 6.4.3. $\mathbf{P}(\mathbf{x}, \mathcal{K})$ *is* 1*-Lipschitz,*

Using the above lemma, we claim that $\hat{G} \leq 2G$. Therefore, using Lemma 6.2.1, $\|\nabla f_t^i(\mathbf{x}_t^i)\| = \|\nabla \hat{f}_t(\mathbf{x}_t^i))\| \leq 2G$. □

Proof of Lemma 6.4.3. Observe the following.

$$\begin{aligned}
\mathbf{P}(\mathbf{x}, \mathcal{K}) - \mathbf{P}(\mathbf{y}, \mathcal{K}) &= \|\mathbf{x} - \Pi_{\mathcal{K}}(\mathbf{x})\| - \|\mathbf{y} - \Pi_{\mathcal{K}}(\mathbf{y})\| \\
&\leq \|\mathbf{x} - \Pi_{\mathcal{K}}(\mathbf{y})\| - \|\mathbf{y} - \Pi_{\mathcal{K}}(\mathbf{y})\| && \Pi_{\mathcal{K}}(\mathbf{y}) \in \mathcal{K} \\
&\leq \|\mathbf{x} - \mathbf{y}\| && \text{triangle inequality}
\end{aligned}$$

□

Proof of Theorem 6.4.2. We give a proof for completeness. This is proved by induction on t.

Induction base. For $t = 1$, we have

$$h_2 \leq h_1(1 - \eta_1) + \eta_1^2 c = c \leq 4c$$

Induction step.

$$\begin{aligned}
h_{t+1} &\leq (1 - \eta_t)h_t + \eta_t^2 c \\
&\leq \left(1 - \frac{2}{t}\right)\frac{4c}{t} + \frac{4c}{t^2} && \text{induction hypothesis} \\
&= \frac{4c}{t}\left(1 - \frac{1}{t}\right) \\
&\leq \frac{4c}{t} \cdot \frac{t}{t+1} && \tfrac{t-1}{t} \leq \tfrac{t}{t+1} \\
&= \frac{4c}{t+1}
\end{aligned}$$

□

6.5 Boosting for bandit linear optimization

Recall that the setting of Bandit Linear Optimization (BLO) is exactly the same as OCO, but (1) the cost functions are linear, and (2) the feedback for the decision maker is the cost it incurs (no gradient information). We use the notation that our linear cost function at time t is:

$$f_t(\mathbf{x}) = f_t^\top \mathbf{x} \ , \ f_t \in \mathbb{R}^d.$$

To boost in the linear bandit setting, we apply techniques for unbiased gradient estimation. Namely, we use randomness to create a random linear function whose expectation is the actual linear function.

We can then hope to use the previous algorithm on the random linear functions and obtain a probabilistic regret bound.

For the rest of this section, we assume the following,

Assumption 6.5.1. *The convex decision set $\mathcal{K}$ contains the unit simplex $K \supseteq \Delta_d$.*

This assumption is very unrestrictive: by scaling and rotation, any set $\mathcal{K}$ which is non-degenerate contains the simplex. Scaling changes regret by a constant factor, and thus we scale so as to contain the unit simplex for notational convenience. An axis-aligning rotation does not affect the regret bound. We use standard basis aligned simplex for notational simplicity.

We can now use standard randomized exploration to create an unbiased estimator of the loss function, as per the following scheme. Let b_t be a Bernoulli random variable with parameter η. Let i_t be chosen uniformly at random from $i_t \in \{1, 2, ..., d\}$.

$$\tilde{f}_t(i) = \begin{cases} 0 & i \neq i_t \\ 0 & i = i_t, b_t = 0 \\ \frac{d}{\eta} \times f_t(i_t) & i = i_t, b_t = 1 \end{cases} \tag{6.5.1}$$

Clearly the expectation of the random vector $\tilde{f}_t$ equals $\mathbf{E}[\tilde{f}_t] = f_t$. We can now use this random variable as feedback for Algorithm 11, as given in Algorithm 12.

Algorithm 12 BoBLO = Boosting Bandit Linear Opt.

```
Input: parameters η, Algorithm 11 instance A.
for t = 1 to T do
    Observe context c_t.
    Draw random Bernoulli b_t w. parameter η.
    if b_t = 1 then
        Pick a coordinate basis i_t ∈ R^d vector randomly.
        Play x_t = i_t.
        Pass f̃_t as per Equation 6.5.1 to A.
    else
        Play x_t = A(c_t), and pass the 0 loss vector to A.
    end if
end for
```

Notice that Algorithm 12 calls upon weak learners for OCO with full information. For this algorithm, our main performance guarantee is as follows. The resulting bound is sub-linear whenever the full-information regret bound is sublinear. Therefore, boosting in bandit linear settings is feasible whenever the full-information version is feasible, albeit at a slower rate.

Theorem 6.5.2 (Boosting for Bandit Linear Optimization)**.** *The predictions* $\mathbf{x}_t$ *generated by Algorithm 12 satisfy*

$$\mathbb{E}\left[\sum_{t=1}^{T} f_t(\mathbf{x}_t)\right] - \inf_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t)) \leq GD\sqrt{T\left(\frac{16dT}{\gamma\sqrt{N}} + \frac{8d}{\gamma}Regret_T\left(\mathcal{W}\right)\right)}$$

Proof. Let $\mathbf{y}_t = \mathcal{A}(c_t)$. Observe that

$$\begin{aligned}
\mathbf{E}\left[\sum_{t=1}^{T} f_t(\mathbf{x}_t)\right] - \inf_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t)) &\leq \mathbf{E}\left[\sum_{t=0}^{T} (\widetilde{f}_t(\mathbf{y}_t) - \widetilde{f}_t(\mathbf{x}_t^\star))\right] + \eta GDT \\
&\leq \frac{4D\tilde{G}T}{\gamma\sqrt{N}} + \frac{2D\tilde{G}}{\gamma^2}\mathrm{Regret}_T(\mathcal{W}) + \eta GDT \qquad \text{Theorem 6.3.1} \\
&\leq \frac{4DdGT}{\eta\gamma\sqrt{N}} + \frac{2dDG}{\eta\gamma}\mathrm{Regret}_T(\mathcal{W}) + \eta GDT
\end{aligned}$$

Above, we use $\tilde{G}$ for an upper bound on the gradients of $\tilde{f}_t$. Lastly, by construction the gradients of the loss function $\tilde{f}_t$ are bounded by $\frac{dG}{\eta}$. Balancing η concludes the claim □

6.6 Boosting for stochastic contextual optimization

In this section we give an alternative viewpoint of our boosting results from the lens of stochastic contextual optimization. The mathematical development is analogous to the previous results, but the statement of results may be of independent interest in statistical learning theory and contextual learning.

Let $\mathcal{K} \subseteq \mathbb{R}^d$ be the (convex) decision set, and $\mathcal{C}$ be the set of possible contexts. In the Stochastic Contextual Optimization problem the aggregate loss attributable to a hypothesis is the expected cost of its actions with respect to a joint distribution $\mathcal{D}$ over the context set $\mathcal{C}$ and the set of convex cost functions over the decision set $\mathcal{K}$.

$$F_{\mathcal{D}}(h) = \mathop{\mathbf{E}}_{(f,c)\sim\mathcal{D}}[f(h(c))].$$

Instead of explicit optimization over the hypothesis class, we assume that we have access to experts that are approximately competitive with the base hypothesis class. Formally, we define a weak optimizer as:

Definition 6.6.1. *A* γ**-weak contextual optimizer** *is a learning algorithm that, when given samples from a distribution* $\mathcal{D}$ *over unit linear loss functions and contexts, outputs a mapping*

$\mathcal{W} : \mathcal{C} \to \mathcal{K}$ *such that*

$$F_{\mathcal{D}}(\mathcal{W}) \leq \gamma \cdot \min_{h \in \mathcal{H}} F_{\mathcal{D}}(h) + \varepsilon.$$

Typically ε scales as $\sim \frac{1}{\sqrt{m}}$ when the learner is given m samples from the distribution $\mathcal{D}$.

The results in this section do **not** follow blackbox from the previous ones. While online boosting algorithms operate in more permissive settings than statistical ones, as they do not require the context and loss vectors to be sampled i.i.d., the corresponding assumption on the weak learners for the online setting are stronger too.

Algorithm 13 below makes use of N weak optimiziers, and generates a hypothesis $h : \mathcal{C} \to \mathcal{K}$ such that the following theorem holds. Notice that the resulting guarantee delivers a solution which is stronger in two aspects than the weak optimization oracle:

1. The resulting solution competes with the convex hull of $\mathcal{H}$, rather than the best fixed $h \in \mathcal{H}$.

2. The resulting solution removes the γ approximation factor of the weak optimizers.

Theorem 6.6.2 (Main-SCO)**.** *The hypothesis h generated by Algorithm 13 with parameter $\delta = \sqrt{\frac{D^2}{\gamma N}}$ satisfy*

$$F_{\mathcal{D}}(h) - \inf_{h^\star \in CH(\mathcal{H})} F_{\mathcal{D}}(h^\star) \leq \frac{4GD}{\gamma\sqrt{N}} + \frac{2GD}{\gamma}\varepsilon.$$

Algorithm 13 B4CO = Boosting for Contextual Opt.

Input: γ-weak contextual optimizer $\mathcal{O}$, parameters $\{\eta_1, \dots \eta_N\}$, δ, $\kappa = G$, distribution $\mathcal{D}$.
Choose h^0 arbitrarily from $\mathcal{H}$.
for $i = 1$ **to** N **do**
    Define a distribution $\mathcal{D}^i$ specified via the following sampling oracle:
        Sample $(f, c) \sim \mathcal{D}$, and define $\hat{f} = X_{\mathcal{K},\delta,\kappa}[f]$
        Define a linear loss function

$$f^i(\mathbf{x}) = \nabla \hat{f}(h^{i-1}(c)) \cdot \mathbf{x}$$

        Output (f^i, c).
    Call a weak optimizer on the distribution $\mathcal{D}^i$ to get a hypothesis $\mathcal{W}^i$.
    Define $h^i = (1 - \eta_i)h^{i-1} + \frac{\eta_i}{\gamma}\mathcal{W}^i$
end for
Return predictor h, defined as $h(c) = \Pi_{\mathcal{K}}(h^N(c))$.

As before, for a convex loss function f, define $\hat{f} = X_{\mathcal{K},\delta,G}[f]$, and note that $\hat{f}_t$ is $\frac{1}{\delta}$-smooth. Notations we use below:

1. $F(h) = \mathbf{E}_{(f,c)\sim\mathcal{D}}[f(h(c))]$

2. $\hat{F}(h) = \mathbf{E}_{(f,c)\sim\mathcal{D}}[\hat{f}(h(c))]$

Lemma 6.6.3. *Suppose, for any $f \in \mathcal{F}$, $\hat{f}$ is β-smooth, and $\max_{f\in\mathcal{F},x\in\mathcal{K}} \|\nabla \hat{f}(\mathbf{x})\| \leq \hat{G}$. Then,*

$$\hat{F}(h) \ - \hat{F}(h^\star) \leq \frac{2\beta D^2}{\gamma^2 N} + \frac{\hat{G}D}{\gamma}\varepsilon.$$

Proof. Let $h^* = \text{argmin}_{h\in CH(\mathcal{H})} F(h)$. Denote for all $i = 0, 1, 2, \ldots, N$, $\Delta_i = \hat{F}(h^i) - \hat{F}(h^\star)$.

Recall that $\hat{f}_t$ is β smooth by our assumption. Thus,

$$\begin{aligned}
\Delta_i &= \hat{F}(h^i) - \hat{F}(h^\star) \\
&= \mathop{\mathbf{E}}_{f,c\sim\mathcal{D}}[\hat{f}(h^i(c))] - \hat{F}(h^\star) \\
&= \mathop{\mathbf{E}}_{f,c\sim\mathcal{D}}\left[\hat{f}(h^{i-1}(c) + \eta_i(\frac{1}{\gamma}\mathcal{W}^i(c) - h^{i-1}(c))\right] - F(h^\star) \\
&\leq \mathop{\mathbf{E}}_{f,c\sim\mathcal{D}}[\hat{f}_t(h^{i-1}(c)) + \eta_i\nabla\hat{f}(h^{i-1}(c))\cdot(\frac{1}{\gamma}\mathcal{W}^i(c) - h^{i-1}(c)) + \frac{\eta_i^2\beta}{2}\|\frac{1}{\gamma}h^{i-1}(c) - \mathcal{W}^i(c)\|^2] - \hat{F}(h^\star) \\
&\leq \mathop{\mathbf{E}}_{f,c\sim\mathcal{D}}[\hat{f}(h^{i-1}(c)) + \eta_i(f^i(\frac{1}{\gamma}\mathcal{W}^i(c)) - f_t^i(h^{i-1}(c))) + \frac{\eta_i^2\beta D^2}{2\gamma^2}] - \hat{F}(h^\star)
\end{aligned}$$

Using linearity of f^i, which implies that optimal aggregate loss values are equal over $\mathcal{H}$ and $CH(\mathcal{H})$ hypotheses classes, and the definition of weak learning, we have

$$\begin{aligned}
\Delta_i &\leq \Delta_{i-1} + \eta_i \mathop{\mathbf{E}}_{f,c\sim\mathcal{D}}\left[\nabla\hat{f}_t(h^{i-1}(c))\cdot(h^\star(c) - h^{i-1}(c))\right] + \frac{\eta_i^2\beta D^2}{2\gamma^2} + \eta_i\frac{\hat{G}D\varepsilon}{\gamma} \\
&\leq \Delta_{i-1}(1-\eta_i) + \frac{\eta_i^2\beta D^2}{2\gamma^2} + \eta_i\frac{\hat{G}D\varepsilon}{\gamma}
\end{aligned}$$

where the last step follows from the convexity of f. We thus have the recurrence

$$\Delta_i \leq \Delta_{i-1}(1-\eta_i) + \eta_i^2\frac{\beta D^2}{2\gamma^2} + \eta_i\frac{\hat{G}D\varepsilon}{\gamma}.$$

As before, substituting $\hat{\Delta}_i = \Delta_i - \frac{\hat{G}D\varepsilon}{\gamma}$, and applying Lemma 6.4.2, gives us $\hat{\Delta}_N \leq \frac{2\beta D^2}{\gamma^2 N}$. □

Proof of Theorem 6.6.2. Using Lemma 6.2.3, we have,

$$\begin{aligned}
F(h) \ - F(h^\star) &= \mathop{\mathbf{E}}_{f,c}[f(h(c))] \ - \mathop{\mathbf{E}}_{f,c}[f(h^\star(c))] \\
&\leq \mathop{\mathbf{E}}_{f,c}[\hat{f}(h(c))] \ - \mathop{\mathbf{E}}_{f,c}[\hat{f}(h^\star(c))] + \delta G^2 \\
&\leq \mathop{\mathbf{E}}_{f,c}[\hat{f}(h^N(c))] \ - \mathop{\mathbf{E}}_{f,c}[\hat{f}(h^\star(c))] + 2\delta G^2
\end{aligned}$$

Boston dataset						
	WL	N=2	N=3	N=4	N=5	Improvement
Decision Stumps	1.000	0.941	0.891	0.853	**0.821**	**17.9%**
Ridge Regression	1.000	**0.980**	0.985	1.009	1.060	**2.0%**
Tiny MLP	1.000	0.987	0.966	0.934	**0.920**	**8.0%**
Diabetes dataset						
	WL	N=2	N=3	N=4	N=5	Improvement
Decision Stumps	1.000	0.975	0.960	0.952	**0.941**	**15.9%**
Ridge Regression	1.000	0.986	0.974	0.965	**0.956**	**4.4%**
Tiny MLP	1.000	0.982	0.981	0.987	**0.965**	**3.5%**
California Housing dataset						
	WL	N=2	N=3	N=4	N=5	Improvement
Decision Stumps	1.000	0.969	0.946	0.932	**0.922**	**7.8%**
Ridge Regression	**1.000**	1.019	1.041	1.066	1.094	**-1.9%**
Tiny MLP	1.000	0.860	0.861	0.888	**0.835**	**16.5%**

Figure 6.1: Performance of the boosting algorithm on 3 different datasets, averaged across 20 runs, as a function of the weak learning class and the number of weak learners. The entries indicate normalized loss value with the base/weak learner loss set to one.

By Lemma 6.2.1, that $\hat{f}$ is $\frac{1}{\delta}$-smooth. By applying Lemma 6.6.3:

$$F(\mathcal{A}) \ - F(h^\star) \leq 2\delta G^2 + \frac{2D^2}{\delta\gamma^2 N} + \frac{\hat{G}D}{\gamma}\varepsilon$$

Because $\mathbf{P}(x, K)$ is 1-Lipschitz, using Lemma 6.2.1, $\|\nabla f_t^i(x_t^i)\| = \|\nabla \hat{f}_t(x_t^i))\| \leq 2G$. Choosing the suggested value of δ yields the proof. □

6.7 Experimental results

While the primary contribution of this work is theoretical, empirically testing our proposal serves as a sanity check for the theoretical results. Also, as stated in the related work, the proposed algorithm is the first of its kind in terms of being able to simultaneously operate with general convex losses and sets, and with multiplicatively approximate (inexact) weak learners. Consequently, the role of experiments here is confirmatory, and is not aimed at achieving state-of-the-art on the considered datasets.

To validate our results, we check if the proposed algorithm is indeed capable of boosting the accuracy of concrete instantiations of weak learners. Three online weak learners were considered: decision stumps, ridge regression, and a tiny multi-layer perceptron with one hidden unit trained via online gradient descent. The implementation for each was suitably adapted from Scikit-Learn [PVG+11].

For each choice of weak learner class, we considered the performance of the boosting algorithm (Algorithm 11) across multiple rounds of boosting or number of weak learners; the computational burden of the algorithm scales linearly with the latter. We evaluated the weak learners and the boosting algorithm on 3 publicly available datasets. The Diabetes dataset used here is present in UCI ML Repository. for regression with square loss, averaging each across 20 runs. The step size η was set to 0.01, and γ was chosen to be 0.1 – these values were not tuned. We present the results in Table 6.1. Further experimental details and code are detailed in the supplement.

The results demonstrate the boosting for online convex optimization does indeed succeed in boosting the accuracy while using only few weak learners (equivalently, within a few rounds of boosting), and therefore, with a reasonably small increase in computation. The improvement is most pronounced for decision stumps, since this is an especially impoverished base class. Also, note that a convex combination of linear models is linear. Therefore, boosting a linear model (like ridge regression) has no expressivity benefit. Mirroring this, the improvements in the ridge regression setting are least prominent, and in some cases negative.

Chapter 7

Provably Efficient Maximum Entropy Exploration

Suppose an agent is in a (possibly unknown) Markov Decision Process in the absence of a reward signal, what might we hope that an agent can efficiently learn to do? This chapter studies a broad class of objectives that are defined solely as functions of the state-visitation frequencies that are induced by how the agent behaves. For example, one natural, intrinsically defined, objective problem is for the agent to learn a policy which induces a distribution over state space that is as uniform as possible, which can be measured in an entropic sense. We provide an efficient algorithm to optimize such such intrinsically defined objectives, when given access to a black box planning oracle (which is robust to function approximation). Furthermore, when restricted to the tabular setting where we have sample based access to the MDP, the proposed algorithm is provably efficient, both in terms of its sample and computational complexities. Key to the algorithmic methodology here is utilizing the conditional gradient method (a.k.a. the Frank-Wolfe algorithm) which utilizes an approximate MDP solver.

7.1 Introduction

A fundamental problem in reinforcement learning is that of exploring the state space. How do we understand what is even possible in the context of a given environment in the absence of a reward signal?

This question has received a lot of attention, with approaches such as learning with intrinsic

reward and curiosity driven methods, surveyed below. This chapter studies a class of objectives that is defined solely as function of the state-visitation frequencies. A natural such objective is finding a policy that maximizes the entropy of the induced distribution over the state space. More generally, this approach extends to any concave function over distributions. In contrast to scalar rewards, such objectives permit a broader framework for reward specifiction, and may be useful in other contexts.

Suppose the MDP is fully and precisely known, in terms of states, actions, and the entire transition matrix. Then maximizing the entropy can be recast as a convex optimization problem (see Section 7.3.2 or [DFVR03]) over the space of state-visitation frequencies induced by the exhaustive set of all policies. However, most RL instances that are common in practice exhibit at least one of several complications:

— prohibitively large state space (i.e. Chess or Go)

— unknown transition matrix (as in common Atari games)

These scenarios often require function approximation, ie. restricting the search to a non-linearly parameterized policy class (eg. neural networks), which makes the entropy maximization problem non-convex.

As a remedy for the computational difficulty, we propose considering an approximate planning oracle: an efficient method that given a well-specified reward signal can find an optimizing policy. Such sample-based planning oracles have been empirically observed to work well with non-linearly parameterized policy classes. Given such an oracle, we give a provably efficient method for exploration based on the conditional gradient (or Frank-Wolfe) algorithm [FW56].

Formally, we show how to generate a sequence of reward signals, that when sequentially optimized give rise to a policy with an entropy on the state distribution close to optimal. The main theorem gives a bound on the number of calls to the planning oracle, which is independent of the size of the state space of the MDP and that of the policy class. Next, we outline an efficient construction of such oracles and state the resultant sample & computational complexity in the tabular MDP setting. As a proof of concept, we implement the proposed method and demonstrate experiments over several mainstream RL tasks in Section 7.6.

7.1.1 Informal statement of results

To facilitate exploration in potentially unknown MDPs within a restricted policy class Π, we assume access to the environment using the following two oracles:

Approximate planning oracle: Given a reward function (on states) $r : \mathcal{S} \to \mathbb{R}$ and a sub-

optimality gap ε, the planning oracle returns a policy $\pi = \textsc{ApproxPlan}(r, \varepsilon)$ with the guarantee that $V(\pi) \geq \max_{\pi \in \Pi} V(\pi) - \varepsilon$, where $V(\pi)$ is the value of policy π.

State distribution estimate oracle: A state distribution oracle estimates the state distribution $\hat{d}_\pi = \textsc{DensityEst}(\pi, \varepsilon)$ of any given (non-stationary) policy π, guaranteeing that $\|d_\pi - \hat{d}_\pi\|_\infty \leq \varepsilon$.

Given access to these two oracles, we describe a method that provably optimizes any continuous and smooth objective over the state-visitation frequencies. Of special interest is the maximum entropy and relative entropy objectives.

Theorem 7.1.1 (Main Theorem - Informal)**.** *There exists an efficient algorithm (Algorithm 14) such that for any β-smooth measure R, and any $\varepsilon > 0$, in $O(\frac{1}{\varepsilon} \log \frac{1}{\varepsilon})$ calls to* ApproxPlan *&* DensityEst *, it returns a policy $\bar{\pi}$ with*

$$R(d_{\bar{\pi}}) \geq \max_{\pi \in \Pi} R(d_\pi) - \varepsilon \,.$$

7.1.2 Related work

We review related works in this section.

Reward Shaping & Imitation Learning: Direct optimization approaches to RL (such as policy gradient methods) tend to perform favorably when random sequences of actions lead the agent to some positive reward, but tend to fail when the rewards are sparse or myopic. Thus far, the most practical approaches to address this have either been through some carefully constructed reward shaping (e.g. [NHR99] where dense reward functions are provided to make the optimization problem more tractable) or through imitation learning [AN04, RGB11] (where an expert demonstrates to the agent how to act).

PAC-RL Learning: For the case of tabular Markov decision processes, the balance of exploration and exploitation has been addressed in that there are a number of methods which utilize confidence based reward bonuses to encourage exploration in order to ultimately behave near optimally [KS02, Kak03, SLW$^+$06, LH14, DB15, SS10, AOM17]. [LA12] offer a Dijkstra-like algorithm for discovering incrementally reachable states in the tabular setting.

Count-based Models & Directed Exploration: There are a host of recent empirical success using deep RL methods which encourage exploration in some form [MKS$^+$15, SHM$^+$16]. The approaches are based on a few related ideas: that of encouraging exploration through state visitation frequencies (e.g. [OBvdOM17, BSO$^+$16, THF$^+$17]) and those based on a intrinsic reward signal derived from novelty or prediction error [LLTyO12, PAED17, SRM$^+$18, FCRL17, MJR15, HCC$^+$16,

WRR+17], aligning an intrinsic reward to the target objective [Kae93, CBS05, SLBS10, SLB09, ZOS18], or sample based approaches to tracking of value function uncertainty [OBPVR16, OAC18].

Intrinsic Learning: Works in [CBS05, SLB09, SLBS10] established computational theories of intrinsic reward signals (and how it might help with downstream learning of tasks) and other works also showed how to incorporate intrinsic rewards (in the absence of any true reward signal) [WdWK+18, BESK18, BEP+18, NPD+18]. The potential benefit is that such learning may help the agent reach a variety of achievable goals and do well on other extrinsically defined tasks, not just the task under which it was explicitly trained for under one specific reward function (e.g. see [CBS05, SLB09, WdWK+18, NPD+18, ABB+]).

7.2 Preliminaries

Markov decision process: An infinite-horizon discounted Markov Decision Process is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \gamma, d_0)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, and d_0 is the distribution of the initial state s_0. At each timestep t, upon observing the state s_t, the execution of action a_t triggers an observable reward of $r_t = r(s_t, a_t)$ and a transition to a new state $s_{t+1} \sim P(\cdot|s_t, a_t)$. The performance on an infinite sequence of states and actions (hereafter, referred to as a *trajectory*) is judged through the (discounted) cumulative reward it accumulates, defined as

$$V(\tau = (s_0, a_0, s_1, a_1, \dots)) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t).$$

Policies: A policy is a (randomized) mapping from a history, say $(s_0, a_0, r_0, s_1, a_1, r_1 \dots s_{t-1}, a_{t-1}, r_{t-1})$, to an action a_t. A stationary policy π is a (randomized) function which maps a state to an action in a time-independent manner, i.e. $\pi : \mathcal{S} \to \Delta(\mathcal{A})$. When a policy π is executed on some MDP $\mathcal{M}$, it produces a distribution over infinite-length trajectories $\tau = (s_0, a_0, s_1, a_1 \dots)$ as specified below.

$$P(\tau|\pi) = P(s_0)\prod_{i=0}^{\infty}(\pi(a_i|s_i)P(s_{i+1}|s_i, a_i))$$

The (discounted) value V_π of a policy π is the expected cumulative reward an action sequence sampled from the policy π gathers.

$$V_\pi = \mathop{\mathbb{E}}_{\tau\sim P(\cdot|\pi)} V(\tau) = (1-\gamma) \mathop{\mathbb{E}}_{\tau\sim P(\cdot|\pi)} \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$$

Induced state distributions: The t-step state distribution and the (discounted) state distribu-

tion of a policy π that result are

$$d_{t,\pi}(s) = P(s_t = s|\pi) = \sum_{\text{all } \tau \text{ with } s_t=s} P(\tau|\pi), \tag{7.2.1}$$

$$d_{t,\pi}(s,a) = P(s_t = s, a_t = a|\pi) = \sum_{\text{all } \tau \text{ with } s_t=s, a_t=a} P(\tau|\pi), \tag{7.2.2}$$

$$d_\pi(s) = (1-\gamma)\sum_{t=1}^{\infty} \gamma^t d_{t,\pi}(s), \tag{7.2.3}$$

$$d_\pi(s,a) = (1-\gamma)\sum_{t=1}^{\infty} \gamma^t d_{t,\pi}(s,a). \tag{7.2.4}$$

The latter distribution can be viewed as the analogue of the stationary distribution in the infinite horizon setting.

Mixtures of stationary policies: Given a sequence of k policies $C = (\pi_0, \dots \pi_{k-1})$, and $\alpha \in \Delta_k$ (the simplex), we define $\pi_{\text{mix}} = (\alpha, C)$ to be a mixture over stationary policies. The (non-stationary) policy π_{mix} is one where, at the first timestep $t = 0$, we sample policy π_i with probability α_i and then use this policy for all subsequent timesteps. In particular, the behavior of a mixture π_{mix} with respect to an MDP is that it induces infinite-length trajectories $\tau = (s_0, a_0, s_1, a_1 \dots)$ with the probability law :

$$P(\tau|\pi_{\text{mix}}) = \sum_{i=0}^{k-1} \alpha_i P(\tau|\pi_i) \tag{7.2.5}$$

and the induced state distribution is:

$$d_{\pi_{\text{mix}}}(s) = \sum_{i=0}^{k-1} \alpha_i d_{\pi_i}(s). \tag{7.2.6}$$

Note that such a distribution over policies need not be representable as a stationary stochastic policy (even if the π_i's are stationary) due to that the sampled actions are no longer conditionally independent given the states.

7.3 The Objective: MaxEnt Exploration

As each policy induces a distribution over states, we can associate a *concave* reward functional $R(\cdot)$ with this induced distribution. We say that a policy π^* is a *maximum-entropy exploration policy*, also to referred to as the *max-ent* policy, if the corresponding induced state distribution has the maximum possible $R(d_\pi)$ among the policy class Π. When considering the class of all policies, Lemma 7.3.3

assures us that the search over the class of stationary policies is sufficient.

$$\pi^* \in \arg\max_{\pi \in \Pi} R(d_\pi).$$

Our goal is to find a policy that induces a state distribution with a comparable value of the reward functional.

7.3.1 Examples of reward functionals

A possible quantity of interest that serves as a motivation for considering such functionals is the entropy of the induced distribution[1].

$$\max_{\pi \in \Pi} \{H(d_\pi) = - \mathop{\mathbb{E}}_{s \sim d_\pi} \log d_\pi(s)\}$$

The same techniques we derive can also be used to optimize other entropic measures. For example, we may be interested in minimizing:

$$\min_{\pi \in \Pi} \left\{ \mathrm{KL}(d_\pi || Q) = \mathop{\mathbb{E}}_{s \sim d_\pi} \log \frac{d_\pi(s)}{Q(s)} \right\}$$

for some given distribution $Q(s)$. Alternatively, we may seek to minimize a cross entropy measure:

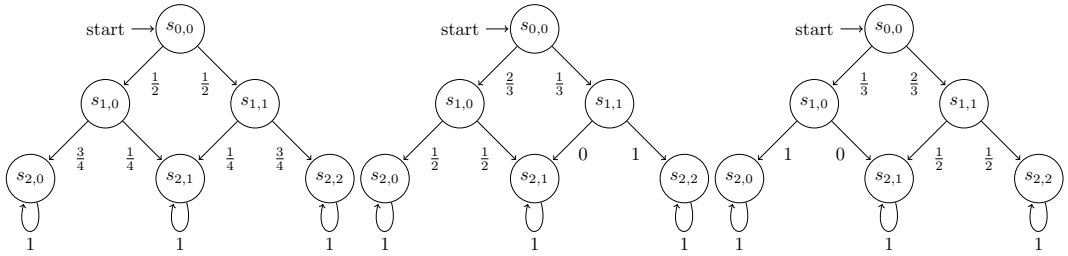
$$\min_{\pi \in \Pi} \left\{ \mathop{\mathbb{E}}_{s \sim Q} \log \frac{1}{d_\pi(s)} = \mathrm{KL}(Q || d_\pi) + H(Q) \right\}$$

where the expectation is now under Q. For uniform Q, this latter measure may be more aggressive in forcing π to have more uniform coverage than the entropy objective.

7.3.2 Landscape of the objective function

In this section, we establish that the entropy of the state distribution is *not* a concave function of the policy. Similar constructions can establish analogous statements for other non-trivial functionals. Subsequently, when the policy class in consideration is exhasutive, we discuss a possible convex reformulation of the objective in the space of induced distributions which constitute a convex set.

[1] Please note the distinction from the conditional entropy of actions given the state, e.g. [Tod07, HZAL18].



(a) π_0, $d_{2,\pi_0} = \left(\frac{3}{8}, \frac{1}{4}, \frac{3}{8}\right)$ (b) π_1, $d_{2,\pi_1} = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ (c) π_2, $d_{2,\pi_2} = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$

Figure 7.1: Description of π_0, π_1, π_2.

Non-convexity in the policy space

Despite the concavity of the entropy function, our overall maximization problem is not concave as the state distribution is not an affine function of the policy. This is stated precisely in the following lemma.

Lemma 7.3.1. *$H(d_\pi)$ is not concave in π.*

Proof. Figure 7.1 demonstrates the behavior of π_0, π_1, π_2 on a 6-state MDP with binary actions. Note that for sufficiently large $\gamma \to 1$ and any policy π, the discounted state distribution converges to the distribution on the states at the second timestep, or formally $d_\pi \to d_{2,\pi}$. Now with the realization $\pi_0 = \frac{\pi_1 + \pi_2}{2}$, observe that d_{2,π_0} is not uniform on $\{s_{2,0}, s_{2,1}, s_{2,2}\}$, implying that $H(d_{2,\pi_0}) < \frac{H(d_{2,\pi_1}) + H(d_{2,\pi_2})}{2}$. □

Lemma 7.3.2. *For any policy π and MDP $\mathcal{M}$, define the matrix $P_\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ so that*

$$P_\pi(s', s) = \sum_{a \in \mathcal{A}} \pi(a|s) P(s'|s, a).$$

Then it is true that

1. *P_π is linear in π,*
2. *$d_{t,\pi} = P_\pi^t d_0$ for all $t \geq 0$,*
3. *$d_\pi = (1 - \gamma)(I - \gamma P_\pi)^{-1} d_0$.*

Proof. Linearity of P_π is evident from the definition. (2,3) may be verified by calculation. □

Convexity in the distribution space

Define the set of all induced distributions as $\mathcal{K} = \{d : d(s, a) \geq 0$ and satisfies the constraints stated below$\}$. For every $d \in \mathcal{K}$, it is possible to construct a policy π with $d_\pi = d$, and for every π, $d_\pi \in \mathcal{K}$ holds

[Put14].

$$\sum_a d(s,a) = (1-\gamma)d_0(s) + \gamma \sum_{s',a'} P(s|s',a')d(s',a')$$

For an exhaustive policy class, the search for a max-ent policy can be recast as a convex optimization problem over the space of distributions.

$$\max_{d \in \mathcal{K}} R(d).$$

Although the above reduction is outlined for an exhaustive policy class, similar reductions are possible for linearly-parameterized policy classes [PMA10, NJG17]. These techniques can be extended to the case of MDPs with unknown dynamics [DFVR03].

Sufficiency of Stationary Policies

The set of non-Markovian policies is *richer* than the set of Markov stationary policies in terms of the distributions over trajectories each may induce. A priori, it is not evident that maximizing $R(d_\pi)$ over the set of stationary policies is sufficient to guarantee the optimality in a larger class of all policies. Lemma 7.3.3 establishes this claim by equating the set of achievable *induced state distributions* for these two sets of policies.

Lemma 7.3.3. *[Put14] For any possibly non-Markovian policy π, define a stationary Markov policy π' as $\pi'(a|s) = \frac{d_\pi(s,a)}{d_\pi(s)}$. Then, $d_\pi = d_{\pi'}$.*

7.4 Algorithms & main results

The algorithm maintains a distribution over policies, and proceeds by adding a new policy to the support of the mixture and reweighing the components. To describe the algorithm, we will utilize access to two kinds of oracles. The constructions for these are detailed in later sections.

Approximate planning oracle: Given a reward function (on states) $r : \mathcal{S} \to \mathbb{R}$ and a sub-optimality gap ε_1, the planning oracle returns a policy[2] $\pi = \textsc{ApproxPlan}(r, \varepsilon_1)$ with the guarantee that $V_\pi \geq \max_{\pi \in \Pi} V_\pi - \varepsilon_1$.

State distribution estimate oracle: A state distribution oracle estimates the state distribution $\hat{d}_\pi = \textsc{DensityEst}(\pi, \varepsilon_0)$ of any given (non-stationary) policy π, guaranteeing that $\|d_\pi - \hat{d}_\pi\|_\infty \leq \varepsilon_0$.

We shall assume in the following discussion that the reward functional R is β-smooth, B-bounded,

[2] As the oracle is solving a discounted problem, we know the optimal value is achieved by a stationary policy.

Algorithm 14 Maximum-entropy policy computation.

Input: Step size η, number of iterations T, planning oracle error tolerance $\varepsilon_1 > 0$, state distribution oracle error tolerance $\varepsilon_0 > 0$, reward functional R.
Set $C_0 = \{\pi_0\}$ where π_0 is an arbitrary policy.
Set $\alpha_0 = 1$.
for $t = 0, \ldots, T-1$ **do**
  Call the state distribution oracle on $\pi_{\mathrm{mix},t} = (\alpha_t, C_t)$:

$$\hat{d}_{\pi_{\mathrm{mix},t}} = \textsc{DensityEst}\left(\pi_{\mathrm{mix},t}, \varepsilon_0\right)$$

  Define the reward function r_t as

$$r_t(s) = \nabla R(\hat{d}_{\pi_{\mathrm{mix},t}}) := \left.\frac{dR(X)}{dX}\right|_{X=\hat{d}_{\pi_{\mathrm{mix},t}}} .$$

  Compute the (approximately) optimal policy on r_t:

$$\pi_{t+1} = \textsc{ApproxPlan}\left(r_t, \varepsilon_1\right) .$$

  Update $\pi_{\mathrm{mix},t+1} = (\alpha_{t+1}, C_{t+1})$ to be

$$C_{t+1} = (\pi_0, \ldots, \pi_t, \pi_{t+1}), \tag{7.4.1}$$
$$\alpha_{t+1} = ((1-\eta)\alpha_t, \eta). \tag{7.4.2}$$

end for
return $\pi_{\mathrm{mix},T} = (\alpha_T, C_T)$.

and that it satisfies the following inequality for all X, Y.

$$\|\nabla R(X) - \nabla R(Y)\|_\infty \le \beta \|X - Y\|_\infty \tag{7.4.3}$$

$$-\beta \mathbb{I} \preceq \nabla^2 R(X) \preceq \beta \mathbb{I}; \quad \|\nabla R(X)\|_\infty \le B \tag{7.4.4}$$

Theorem 7.4.1 (Main Theorem)**.** *For any $\varepsilon > 0$, set $\varepsilon_1 = 0.1\varepsilon$, $\varepsilon_0 = 0.1\beta^{-1}\varepsilon$, and $\eta = 0.1\beta^{-1}\varepsilon$. When Algorithm 14 is run for T iterations where:*

$$T \ge 10\beta\varepsilon^{-1} \log 10B\varepsilon^{-1} ,$$

we have that:

$$R(d_{\pi_{mix,T}}) \ge \max_{\pi \in \Pi} R(d_\pi) - \varepsilon .$$

Before we begin the proof, we state the implication for maximizing the entropy of the induced distribution. While the entropy objective is, strictly speaking, not smooth, one may consider a smoothed alternative H_σ defined below.

$$H_\sigma(d_\pi) = -\mathbb{E}_{s \sim d_\pi} \log(d_\pi(s) + \sigma)$$

When the algorithm is fed H_σ as the *proxy* reward functional, it is possible make sub-optimality guarantees on the true objective H. The next lamma relates the entropy functional H to its smoothed variant H_σ, while the rest of the lemma quantifies smoothness of H_σ. The factors of $|\mathcal{S}|$ incurred below are a consequence of imposed smoothing on H, and are not necessary for naturally smooth objectives.

The following lemma is helpful in proving the corollary for the entropy functional.

Lemma 7.4.2. *For any two distributions $P, Q \in \Delta_d$:*

(A) $(\nabla H_\sigma(P))_i = -\left(\log(P_i + \sigma) + \frac{P_i}{P_i+\sigma}\right)$,

(B) $H_\sigma(P)$ *in concave in* P,

(C) $H_\sigma(P)$ *is* $2\sigma^{-1}$ *smooth, ie.*

$$-2\sigma^{-1}\mathbb{I}_d \preceq \nabla^2 H_\sigma(P) \preceq 2\sigma^{-1}\mathbb{I}_d,$$

(D) $|H_\sigma(P) - H(P)| \leq d\sigma$,

(E) $\|\nabla H_\sigma(P) - \nabla H_\sigma(Q)\|_\infty \leq 2\sigma^{-1}\|P - Q\|_\infty$.

Proof of Lemma 7.4.2. (A) may be verified by explicit calculation. Observe $\nabla^2 H_\sigma(P)$ is a diagonal matrix with entries

$$(\nabla^2 H_\sigma(P))_{i,i} = -\frac{P_i + 2\sigma}{(P_i + \sigma)^2}.$$

(B) is immediate. (C) follows as $|(\nabla^2 H_\sigma(P))_{i,i}| \leq 2\sigma^{-1}$.

$$|H_\sigma(P) - H(P)| = \sum_{i=0}^{d-1} P_i \log \frac{P_i + \sigma}{P_i} \leq \sum_{i=0}^{d-1} P_i \frac{\sigma}{P_i} = d\sigma.$$

The last inequality follows from $\log x \leq x - 1, \forall x > 0$. Finally, to see (E), using Taylor's theorem, observe

$$\begin{aligned}\|\nabla H_\sigma(P) - \nabla H_\sigma(Q)\|_\infty &\leq \max_{i,\alpha\in[0,1]} |(\nabla^2 H_\sigma(\alpha P + (1-\alpha)Q)_{i,i}| \|P - Q\|_\infty \\ &\leq 2\sigma^{-1}\|P - Q\|_\infty.\end{aligned}$$

□

Corollary 7.4.3. *For any $\varepsilon > 0$, set $\sigma = \frac{0.1\varepsilon}{2|\mathcal{S}|}$, $\varepsilon_1 = 0.1\varepsilon$, $\varepsilon_0 = \frac{0.1\varepsilon^2}{80|\mathcal{S}|}$, and $\eta = \frac{0.1\varepsilon^2}{40|\mathcal{S}|}$. When Algorithm 14 is run for T iterations with the reward functional H_σ, where:*

$$T \geq \frac{40|\mathcal{S}|}{0.1\varepsilon^2} \log \frac{\log |\mathcal{S}|}{0.1\varepsilon},$$

we have that:

$$H(d_{\pi_{mix,T}}) \geq \max_{\pi \in \Pi} H(d_\pi) - \varepsilon.$$

We continue with the proof of the main theorem.

Proof of Theorem 7.4.1. Let π^* be a maximum-entropy policy, ie. $\pi^* \in \arg\max_{\pi \in \Pi} R(d_\pi)$.

$$\begin{aligned} R(d_{\pi_{\mathrm{mix},t+1}}) &= R((1-\eta) d_{\pi_{\mathrm{mix},t}} + \eta d_{\pi_{t+1}}) && \text{Equation 7.2.6} \\ &\geq R(d_{\pi_{\mathrm{mix},t}}) + \eta \langle d_{\pi_{t+1}} - d_{\pi_{\mathrm{mix},t}}, \nabla R(d_{\pi_{\mathrm{mix},t}}) \rangle - \eta^2 \beta \| d_{\pi_{t+1}} - d_{\pi_{\mathrm{mix},t}} \|_2^2 && \text{smoothness} \end{aligned}$$

The second inequality follows from the smoothness[3] of R. To incorporate the error due to the two oracles, observe

$$\begin{aligned} \langle d_{\pi_{t+1}}, \nabla R(d_{\pi_{\mathrm{mix},t}}) \rangle &\geq \langle d_{\pi_{t+1}}, \nabla R(\hat{d}_{\pi_{\mathrm{mix},t}}) \rangle - \beta \| d_{\pi_{\mathrm{mix},t}} - \hat{d}_{\pi_{\mathrm{mix},t}} \|_\infty \\ &\geq \langle d_{\pi^*}, \nabla R(\hat{d}_{\pi_{\mathrm{mix},t}}) \rangle - \beta\varepsilon_0 - \varepsilon_1 \\ &\geq \langle d_{\pi^*}, \nabla R(d_{\pi_{\mathrm{mix},t}}) \rangle - 2\beta\varepsilon_0 - \varepsilon_1 \end{aligned}$$

The first and last inequalities invoke the assumptions laid out in Equation 7.4.3. Note that the second inequality above follows from the defining character of the planning oracle, ie. with respect to the reward vector $r_t = \nabla R(\hat{d}_{\pi_{\mathrm{mix},t}})$, for any policy $\pi' \in \Pi$, it holds true that

$$V_{\pi_{t+1}} = \langle d_{\pi_{t+1}}, r_t \rangle \geq V_{\pi'} - \varepsilon_1 = \langle d_{\pi'}, r_t \rangle - \varepsilon_1$$

In particular, this statement holds[4] for the choice $\pi' = \pi^*$.

[3] See Section 2.1 in [B+15] for equivalent definitions of smoothness in terms of the function value and the Hessian.

[4] Even when when Π is chosen to be set of all stationary policies, this argument does not rely on π^* being a stationary policy, since π_{t+1} is an optimal policy for the reward function r_t among the class of all policies.

Using the above fact and continuing on

$$\begin{aligned} R(d_{\pi_{\mathrm{mix},t+1}}) &\geq R(d_{\pi_{\mathrm{mix},t}}) + \eta\langle d_{\pi^*} - d_{\pi_{\mathrm{mix},t}}, \nabla R(d_{\pi_{\mathrm{mix},t}})\rangle - 2\eta\beta\varepsilon_0 - \eta\varepsilon_1 - \eta^2\beta \\ &\geq (1-\eta)R(d_{\pi_{\mathrm{mix},t}}) + \eta R(d_{\pi^*}) - 2\eta\beta\varepsilon_0 - \eta\varepsilon_1 - \eta^2\beta \end{aligned}$$

The last step here utilizes the concavity of R. Indeed, the inequality follows immediately from the sub-gradient characterization of concave functions. Now, with the aid of the above, we observe the following inequality.

$$R(d_{\pi^*}) - R(d_{\pi_{\mathrm{mix},t+1}}) \leq (1-\eta)(R(d_{\pi^*}) - R(d_{\pi_{\mathrm{mix},t}})) + 2\eta\beta\varepsilon_0 + \eta\varepsilon_1 + \eta^2\beta.$$

Telescoping the inequality, this simplifies to

$$\begin{aligned} R(d_{\pi^*}) - R(d_{\pi_{\mathrm{mix},T}}) &\leq (1-\eta)^T (R(d_{\pi^*}) - R(d_{\pi_{\mathrm{mix},0}})) + 2\beta\varepsilon_0 + \varepsilon_1 + \eta\beta \\ &\leq Be^{-T\eta} + 2\beta\varepsilon_0 + \varepsilon_1 + \eta\beta. \end{aligned}$$

Setting $\varepsilon_1 = 0.1\varepsilon$, $\varepsilon_0 = 0.1\beta^{-1}\varepsilon$, $\eta = 0.1\beta^{-1}\varepsilon$, $T = \eta^{-1}\log 10B\varepsilon^{-1}$ suffices. □

7.4.1 Tabular setting

In general, the construction of provably computationally efficient approximate planning oracle for MDPs with large or continuous state spaces poses a challenge. Discounting limited settings (eg. the Linear Quadratic Regulators [Ber05], [FGKM18]), one may only appeal to the recent empirical successes of sample-based planning algorithms that rely on the power of non-linear function approximation.

Nevertheless, one may expect, and possibly require, that any solution proposed to address the general case performs reasonably when restricted to the tabular setting. In this spirit, we outline the construction of the required oracles in the tabular setting, where we consider the exhaustive class of policies.

The known MDP case

With the knowledge of the transition matrix P of a MDP $\mathcal{M}$ in the form of an explicit tensor, the planning oracle can be implemented via any of the exact solution methods [Ber05], eg. value iteration, linear programming. The state distribution oracle can be efficiently implemented as Lemma 7.3.2

suggests.

Corollary 7.4.4. *When the MDP $\mathcal{M}$ is known explicitly, with the oracles described in Section 7.4, Algorithm 14 runs in* poly $\left(\beta, |\mathcal{S}|, |\mathcal{A}|, \frac{1}{1-\gamma}, \frac{1}{\varepsilon}, \log B\right)$ *time to guarantee* $R(d_{\pi_{mix,T}}) \geq \max_\pi R(d_\pi) - \varepsilon$.

The unknown MDP case

For the case of an unknown MDP, a sample-based algorithm must iteratively try to learn about the MDP through its interactions with the environment. Here, we assume a γ-discounted episodic setting, where the agent can act in the environment starting from $s_0 \sim d_0$ for some number of steps, and is then able to reset. Our measure of sample complexity in this setting is the number of $\tilde{O}\left((1-\gamma)^{-1}\right)$-length episodes the agent must sample to achieve a ε-suboptimal performance guarantee.

The algorithm outlined below makes a distinction between the set of states it is (relatively) sure about and the set of states that have not been visited enough number of times yet. The algorithm and the analysis is similar to the E^3 algorithm [KS02]. Since algorithms like E^3 proceed by building a relatively accurate model on the set of *reachable* states, as opposed to estimate of the value functions, this permits the reuse of information across different invocations, each of which might operate on a different reward signal.

Theorem 7.4.5. *For an unknown MDP, with Algorithm 15 as the planning oracle and Algorithm 16 as the distribution estimate oracle, Algorithm 14 runs in* poly $\left(\beta, |\mathcal{S}|, |\mathcal{A}|, \frac{1}{1-\gamma}, \frac{1}{\varepsilon}\right)$ *time and executes* $\tilde{O}\left(\frac{B^3|\mathcal{S}|^2|\mathcal{A}|}{\varepsilon^3(1-\gamma)^2} + \frac{\beta^3}{\varepsilon^3}\right)$ *episodes of length* $\tilde{O}\left(\frac{\log|\mathcal{S}|\varepsilon^{-1}}{\log\gamma^{-1}}\right)$ *to guarantee that*

$$R(d_{\pi_{mix,T}}) \geq \max_\pi R(d_\pi) - \varepsilon.$$

A sub-optimality bound may be derived on the non-smooth entropy functional H via Lemma A.1. Again, the extraneous factors introduced in the process are a consequence of the imposed smoothing via H_σ.

Corollary 7.4.6. *For an unknown MDP, with Algorithm 15 as the planning oracle and Algorithm 16 as the distribution estimate oracle and H_σ as the* proxy *reward functional, Algorithm 14 runs in* poly $\left(|\mathcal{S}|, |\mathcal{A}|, \frac{1}{1-\gamma}, \frac{1}{\varepsilon}\right)$ *time and executes* $\tilde{O}\left(\frac{|\mathcal{S}|^2|\mathcal{A}|}{\varepsilon^3(1-\gamma)^2} + \frac{|\mathcal{S}|^3}{\varepsilon^6}\right)$ *episodes of length* $\tilde{O}\left(\frac{\log|\mathcal{S}|\varepsilon^{-1}}{\log\gamma^{-1}}\right)$ *to guarantee that*

$$H(d_{\pi_{mix,T}}) \geq \max_\pi H(d_\pi) - \varepsilon.$$

Algorithm 15 Sample-based planning for an unknown MDP.

Input: Reward r, error tolerance $\varepsilon > 0$, exact planning oracle tolerance $\varepsilon_1 > 0$, oversampling parameter m, number of rollouts n, rollout length t_0.

Initialize a persistent data structure $C \in \mathbb{R}^{|\mathcal{S}|^2 \times |\mathcal{A}|}$, which is maintained across different calls to the planning algorithm to keep transition counts, to $C(s'|s,a) = 0$ for every $(s', s, a) \in \mathcal{S}^2 \times \mathcal{A}$.

repeat

Declare $\mathcal{K} = \{s : \min_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} C(s'|s,a) \geq m\}$, $\hat{P}(s'|s,a) = \begin{cases} \frac{C(s'|s,a)}{\sum_{s' \in \mathcal{S}} C(s'|s,a)}, & \text{if } s \in \mathcal{K} \\ \mathbf{1}_{s'=s}. & \text{otherwise.} \end{cases}$

Define the reward function as $r_{\mathcal{K}}(s) = \begin{cases} r(s), & \text{if } s \in \mathcal{K} \\ B. & \text{otherwise} \end{cases}$.

Compute an (approximately) optimal policy on the MDP induced by $\hat{P}$ and reward $r_{\mathcal{K}}$. This task is purely computational, and can be done as indicated in Section 7.4.1. Also, modify the policy so that on every state $s \in \mathcal{S} - \mathcal{K}$, it chooses the least performed action.

$$\pi(s) = \begin{cases} (\Pi(r_{\mathcal{K}}, \varepsilon_1))(s) & \text{if } s \in \mathcal{K}, \\ \operatorname{argmin}_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} C(s'|s,a) & \text{otherwise} \end{cases}$$

Run π on the true MDP $\mathcal{M}$ to obtain n independently sampled t_0-length trajectories $(\tau_1, \ldots \tau_n)$, and increment the corresponding counts in $C(s'|s,a)$.

If and only if no trajectory τ_i contains a state $s \in \mathcal{S} - \mathcal{K}$, mark π as *stable*.

until π is *stable*.

return π.

Algorithm 16 Sample-based estimate of the state distribution.

Input: A policy π, termination length t_0, oversampling parameter m.

Sample m trajectories $(\tau_0, \ldots \tau_{m-1})$ of length t_0 following the policy π.

For every $t < t_0$, calculate the empirical state distribution $\hat{d}_{t,\pi}$.

$$d_{t,\pi}(s) = \frac{|\{i < m : \tau_i = (s_0, a_0, \ldots) \text{ with } s_t = s\}|}{m}$$

return $\hat{d}_\pi = \frac{1-\gamma}{1-\gamma^{t_0}} \sum_{t=0}^{t_0-1} \gamma^t \hat{d}_{t,\pi}$

Before we state the proof, we note the following lemmas. The first is an adaptation of the analysis of the E^3 algorithm. The second is standard. We only include the second for completeness.

Lemma 7.4.7. *For any reward function r with $\|r\|_\infty \leq B$, $\varepsilon > 0$, with $\varepsilon_1 = 0.1B^{-1}\varepsilon, m = \frac{32B^2|\mathcal{S}| \log \frac{2|\mathcal{S}|}{\delta}}{(1-\gamma)^2(0.1\varepsilon)^2}, n = \frac{B \log \frac{32|\mathcal{S}|^2|\mathcal{A}| \log \frac{2|\mathcal{S}|}{\delta}}{(1-\gamma)^2(0.1\varepsilon)^2\delta}}{0.1\varepsilon}, t_0 = \frac{\log \frac{0.1\varepsilon}{\log |\mathcal{S}|}}{\log \gamma}$, Algorithm 7.4.5 guarantees with probability $1 - \delta$*

$$V_\pi \geq \max_\pi V_\pi - \varepsilon.$$

Furthermore, note that if Algorithm 7.4.5 is invoked T times (on possibly different reward functions), the total number of episodes sampled across all the invocations is $n(T + m|\mathcal{S}||\mathcal{A}|) = \tilde{O}\left(\frac{BT}{\varepsilon} + \frac{B^3|\mathcal{S}|^2|\mathcal{A}|}{\varepsilon^3(1-\gamma)^2}\right)$, each episode being of length t_0.

Lemma 7.4.8. *For any $\varepsilon_0, \delta > 0$, when Algorithm 16 is run with $m = \frac{200}{\varepsilon_0^2} \log \frac{2|\mathcal{S}| \log 0.1\varepsilon}{\delta \log \gamma}$, $t_0 = \frac{\log 0.1\varepsilon_0}{\log \gamma}$,*

$\hat{d}_\pi$ satisfies $\|\hat{d}_\pi - d_\pi\|_\infty \le \varepsilon_0$ with probability at least $1-\delta$. In this process, the algorithm samples m episodes of length t_0.

Proof of Theorem 7.4.5. The claim follows immediately from the invocations of the two lemmas above with the parameter settings proposed in Theorem 7.4.1. □

7.5 Proofs for the tabular setting

The following notions & lemmas are helpful in proving Lemma 7.4.7. We shall call a state $s \in \mathcal{K}$ *m-known* if, for all actions $a \in \mathcal{A}$, action a has been executed at state s at least m times. For any MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \gamma)$ and a set of m-known states $\mathcal{K} \subseteq \mathcal{S}$, define an induced MDP $\mathcal{M}_\mathcal{K} = (\mathcal{S}, \mathcal{A}, r_\mathcal{K}, P_\mathcal{K}, \gamma)$ so that the states absent from $\mathcal{K}$ are absorbing and maximally rewarding.

$$r_\mathcal{K}(s,a) = \begin{cases} r(s,a) & \text{if } s \in \mathcal{K}, \\ B & \text{otherwise,} \end{cases} \tag{7.5.1}$$

$$P_\mathcal{K}(s'|s,a) = \begin{cases} P(s'|s,a) & \text{if } s \in \mathcal{K}, \\ \mathbf{1}_{s'=s} & \text{otherwise.} \end{cases} \tag{7.5.2}$$

The state distribution induced by a policy π on $\mathcal{M}_\mathcal{K}$ shall be denoted by $d_{\mathcal{M}_\mathcal{K},\pi}$. Often, in absence of an exact knowledge of the transition matrix P, the policy π may be executed on an estimated transition matrix $\hat{P}$. We shall use $d_{\hat{\mathcal{M}}_\mathcal{K},\pi}$ to denote the state distribution of the policy π executed on the MDP with the transition matrix $\hat{P}$. Also, define the following.

$$P_\mathcal{K}(\text{escape}|\pi) = \mathop{\mathbb{E}}_{\tau\sim P(\cdot|\pi)} \mathbf{1}_{\exists t<t_0 : s_t \notin \mathcal{K}, \tau=(s_0,a_0,\dots)},$$

$$P_{\mathcal{K},\gamma}(\text{escape}|\pi) = (1-\gamma) \mathop{\mathbb{E}}_{\tau\sim P(\cdot|\pi)} \sum_{t=0}^{\infty} \gamma^t \mathbf{1}_{\substack{s_u\in\mathcal{K}\forall u<t \text{ and} \\ s_t\notin\mathcal{K}, \tau=(s_0,a_0,\dots)}}.$$

Note that $P_\mathcal{K}(\text{escape}|\pi) \ge P_{\mathcal{K},\gamma}(\text{escape}|\pi) - \gamma^{t^0}$.

Lemma 7.5.1. *(Lemma 8.4.4[Kak03]) For any policy π, the following statements are valid.*

$$\langle d_\pi, r\rangle \ge \langle d_{\mathcal{M}_\mathcal{K},\pi}, r_\mathcal{K}\rangle - P_{\mathcal{K},\gamma}(\textit{escape}|\pi)\|r_\mathcal{K}\|_\infty,$$

$$\langle d_{\mathcal{M}_\mathcal{K},\pi}, r_\mathcal{K}\rangle \ge \langle d_\pi, r\rangle.$$

Lemma 7.5.2. *(Lemma 8.5.4[Kak03]) If, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, $\|\hat{P}(\cdot|s,a) - P_{\mathcal{K}}(\cdot|s,a)\|_1 \leq \varepsilon$, then for any reward r, policy π, it is true that*

$$|\langle d_{\mathcal{M}_{\mathcal{K}},\pi}, r\rangle - \langle d_{\hat{\mathcal{M}}_{\mathcal{K}},\pi}, r\rangle| \leq \frac{\varepsilon}{1-\gamma}$$

Lemma 7.5.3. *(Folklore, eg. Lemma 8.5.5[Kak03]) When m samples $\{x_1, \ldots x_m\}$ are drawn from a distribution P, supported on a domain of size d, to construct an empirical distribution $\hat{P}(x) = \frac{\sum_{i=1}^{m} \mathbf{1}_{x_i = x}}{m}$, it is guaranteed that with probability $1-\delta$*

$$\|P - \hat{P}\|_1 \leq \sqrt{\frac{8d \log \frac{2d}{\delta}}{m}}.$$

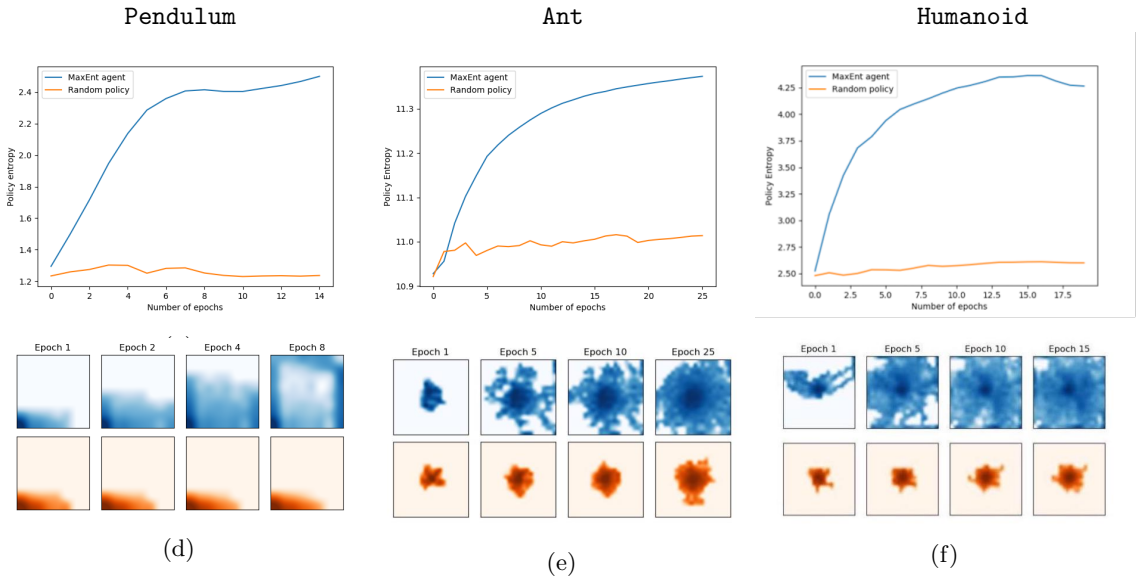
Proof of Lemma 7.4.7. The key observation in dealing with an unknown MDP is: either π, computed on the the transition $\hat{P}$, is (almost) optimal for the given reward signal on the true MDP, or it escapes the set of known states $\mathcal{K}$ quickly. If the former occurs, the requirement on the output of the algorithm is met. In case of the later, π serves as a good policy to quickly explore new states – this can happen only a finite number of times.

Let $\pi^* = \arg\max_\pi V_\pi$. First, note that for any π chosen in the Line 6, we have

$$\begin{aligned}
V_\pi =& \langle d_\pi, r\rangle \\
\geq& \langle d_{\mathcal{M}_{\mathcal{K}},\pi}, r_{\mathcal{K}}\rangle - (\gamma^{t_0} + P_{\mathcal{K}}(\text{escape}|\pi))B && 7.5.1 \\
\geq& \langle d_{\hat{\mathcal{M}}_{\mathcal{K}},\pi}, r_{\mathcal{K}}\rangle - \frac{1}{1-\gamma}\sqrt{\frac{8|\mathcal{S}|\log\frac{2|\mathcal{S}|}{\delta}}{m}} - (\gamma^{t_0} + P_{\mathcal{K}}(\text{escape}|\pi))B && 7.5.2, 7.5.3 \\
\geq& \langle d_{\hat{\mathcal{M}}_{\mathcal{K}},\pi^*}, r_{\mathcal{K}}\rangle - \varepsilon_1 - \frac{1}{1-\gamma}\sqrt{\frac{8|\mathcal{S}|\log\frac{2|\mathcal{S}|}{\delta}}{m}} - (\gamma^{t_0} + P_{\mathcal{K}}(\text{escape}|\pi))B && \text{choice of } \pi \\
\geq& \langle d_{\mathcal{M}_{\mathcal{K}},\pi^*}, r_{\mathcal{K}}\rangle - \varepsilon_1 - \frac{2}{1-\gamma}\sqrt{\frac{8|\mathcal{S}|\log\frac{2|\mathcal{S}|}{\delta}}{m}} - (\gamma^{t_0} + P_{\mathcal{K}}(\text{escape}|\pi))B && 7.5.2, 7.5.3 \\
\geq& V_{\pi^*} - \varepsilon_1 - \frac{2}{1-\gamma}\sqrt{\frac{8|\mathcal{S}|\log\frac{2|\mathcal{S}|}{\delta}}{m}} - (\gamma^{t_0} + P_{\mathcal{K}}(\text{escape}|\pi))B && 7.5.1
\end{aligned}$$

If $P_{\mathcal{K}}(\text{escape}|\pi) > \Delta$, then the probability that π doesn't escape $\mathcal{K}$ in n trials is $e^{-n\Delta}$. Accounting for the failure probabilities with a suitable union bound, Line 8 ensures that π is marked *stable* only if $P_{\mathcal{K}}(\text{escape}|\pi) \leq \frac{\log(N\delta^{-1})}{n}$, where N is the total number of times the inner loop is executed.

To observe the truth of the second part of the claim, note that every reiteration of the inner loop coincides with the exploration of some action at a *m-unknown* state. There can be at most $m|\mathcal{S}||\mathcal{A}|$



(d) (e) (f)

Figure 7.2: Results of the preliminary experiments. In each plot, blue represents the MaxEnt agent, and orange represents the random baseline. 7.2a, 7.2b, and 7.2c show the entropy of the policy evolving with the number of epochs. 7.2d, 7.2e, and 7.2f show the log-probability of occupancy of the two-dimensional state space. In 7.2e and 7.2f, the infinite xy grid is limited to $[-20, 20] \times [-20, 20]$.

such *exploration* steps. Finally, each run of the inner loop samples n episodes. □

Proof of Lemma 7.4.8. First note that it suffices to ensure for all $t < t_0$ simultaneously, it happens $\|d_{t,\pi} - \hat{d}_{t,\pi}\|_\infty \leq 0.1\varepsilon_0$. This is because

$$\|d_\pi - \hat{d}_\pi\|_\infty \leq \frac{1-\gamma}{(1-\gamma^{t_0})} \sum_{t=0}^{t_0-1} \gamma^t \|\hat{d}_{t,\pi} - (1-\gamma^{t_0}) d_{t,\pi}\|_\infty + \gamma^{t_0} \leq \frac{1-\gamma}{(1-\gamma^{t_0})} \sum_{t=0}^{t_0-1} \gamma^t \|\hat{d}_{t,\pi} - d_{t,\pi}\| + 0.3\varepsilon_0 \leq \varepsilon_0.$$

Since the trajectories are independently, $|\hat{d}_{t,\pi}(s) - d_{t,\pi}(s)| \leq \sqrt{\frac{2}{m} \log \frac{2}{\delta}}$ for each $t < t_0$ and state $s \in \mathcal{S}$ with probability $1 - \delta$, by Hoeffding's inequality. A union bound over states and t concludes the proof. □

7.6 Proof-of-concept experiments

We report the results from a preliminary set of experiments. In each case, the MaxEnt agent learns to access the set of reachable states within a small number of iterations, while monotonically increasing the entropy of the induced state distribution.

Recall that Algorithm 14 requires access to an approximate planning oracle and a density estimator for the induced distribution. In most of the experimental environments, the density estimator is deliberately chosen to be simple – a count-based estimate over the discretized state space. It is

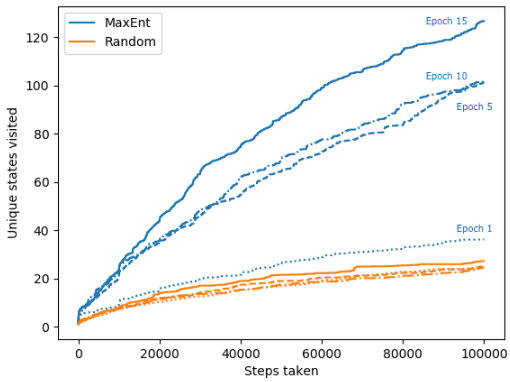


Figure 7.3: The number of distinct xy states visited by `Ant` at various epochs. Results were averaged over $N = 20$ executions. As the number of policies in the mixture increases, the agent reaches more unique states in the same amount of time.

possible to use neural density estimators and other function-approximation based estimators in its stead, as we do for the `Humanoid` environment.

7.6.1 Environments and density estimation

Pendulum. The 2-dimensional state space for `Pendulum` (from [BCP+16]) was discretized evenly to a grid of dimension 8×8. The dimensions represented angular position and angular velocity. For `Pendulum`, the maximum torque and velocity were capped at 1.0 and 7.0, respectively.

Ant. The 29-dimensional state space for `Ant` (with a Mujoco engine) was first reduced to dimension 7, combining the agent's x and y location in the gridspace with a 5-dimensional random projection of the remaining 27 states. The x and y dimensions were discretized into 16 bins in the range $[-12, 12]$. The other dimensions were each normalized and discretized into 15 bins in the range $[-1, 1]$. While the planning agent agent had access to the full state representation, the density estimation was performed exclusively on the reduced representation.

Humanoid. The 376-dimensional state space for the Mujoco `Humanoid` environment was too large and complex to effectively discretize without significantly affecting the accuracy of estimation. In view of this, the agent's state space density was estimated using kernel density estimation (from scikit-learn [PVG+11]), with an Epanechnikov kernel with a bandwidth of 0.10.

7.6.2 Algorithmic details

Reward functional. Each planning agent was trained to maximize a smooth variant of the KL divergence objective.

$$\min_{\pi \in \Pi} \left\{ KL_\sigma(\text{Unif}||d_\pi) = - \mathop{\mathbb{E}}_{s \sim \text{Unif}} \log(d_\pi(s) + \sigma) + C \right\}.$$

Such a choice encourages visitation of novel states more aggressively than the entropy objective. The smoothing parameter was chosen to be $\sigma = |\mathcal{S}|^{-\frac{1}{2}}$.

Pendulum. The planning oracle is a REINFORCE [SMSM00] agent, where the the output policy from the previous iteration is used as the initial policy for the next iteration. The policy class is a neural net with a single hidden layer consisting of 128 units. The agent is trained on 200 episodes every epoch. The baseline agent chooses its action randomly at every time step.

Ant. The planning oracle is a Soft Actor-Critic [HZAL18] agent. The policy class is a neural net with 2 hidden layers composed of 300 units and the ReLU activation function. The agent is trained for 30 episodes, each of which consists of a roll-out of 5000 steps. The mixed policy is executed over 10 trials of $T = 10000$ steps at the end of each epoch in order to approximate the policy distribution and compute the next reward function. The baseline agent chooses its actions randomly for the same number of trials and steps.

Humanoid. The planning oracle is a Soft Actor-Critic agent. The policy class is a neural net with 2 hidden layers composed of 300 units and the ReLU activation function. The agent is trained for 30 episodes, each of which consists of a roll-out of 5000 steps. The mixed policy is executed for 20 trials of $T = 50,000$ steps to collect input data that is used to fit the kernel estimation model. This model is then used to estimate the induced state density of the mixed policy.

Bibliography

[ABB+] Oluwatosin Adewale, Alex Beatson, Davit Buniatyan, Jason Ge, Mikhail Khodak, Holden Lee, Niranjani Prasad, Nikunj Saunshi, Ari Seff, Karan Singh, et al. Pixie: a social chatbot.

[ABC+19] Naman Agarwal, Brian Bullins, Xinyi Chen, Elad Hazan, Karan Singh, Cyril Zhang, and Yi Zhang. Efficient full-matrix adaptive regularization. In *International Conference on Machine Learning*, pages 102–110. PMLR, 2019.

[ABH+19] Naman Agarwal, Brian Bullins, Elad Hazan, Sham Kakade, and Karan Singh. Online control with adversarial disturbances. In *Proceedings of the 36th International Conference on Machine Learning*, pages 111–119, 2019.

[ABHL19] Naman Agarwal, Nataly Brukhim, Elad Hazan, and Zhou Lu. Boosting for dynamical systems. *arXiv preprint arXiv:1906.08720*, 2019.

[ACM07] Hyo-Sung Ahn, YangQuan Chen, and Kevin L Moore. Iterative learning control: Brief survey and categorization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1099–1121, 2007.

[ADT12] Raman Arora, Ofer Dekel, and Ambuj Tewari. Online bandit learning against an adaptive adversary: from regret to policy regret. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1503–1510, 2012.

[AHK12] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[AHK+14] Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*, pages 1638–1646, 2014.

[AHL+18] Sanjeev Arora, Elad Hazan, Holden Lee, Karan Singh, Cyril Zhang, and Yi Zhang. Towards provable control for unknown linear dynamical systems. *International Conference on Learning Representations*, 2018. rejected: invited to workshop track.

[AHM15] Oren Anava, Elad Hazan, and Shie Mannor. Online learning for adversaries with memory: price of past mistakes. In *Advances in Neural Information Processing Systems*, pages 784–792, 2015.

[AHMS21] Naman Agarwal, Elad Hazan, Anirudha Majumdar, and Karan Singh. A regret minimization approach to iterative learning control. *arXiv preprint arXiv:2102.13478*, 2021.

[AHS19] Naman Agarwal, Elad Hazan, and Karan Singh. Logarithmic regret for online control. *arXiv preprint arXiv:1909.05062*, 2019.

[AN04] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[AOM17] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. *arXiv preprint arXiv:1703.05449*, 2017.

[AQN06] Pieter Abbeel, Morgan Quigley, and Andrew Y Ng. Using inaccurate models in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 1–8. ACM, 2006.

[AS17] Naman Agarwal and Karan Singh. The price of differential privacy for online learning. In *International Conference on Machine Learning*, pages 32–40. PMLR, 2017.

[AYBK14] Yasin Abbasi-Yadkori, Peter Bartlett, and Varun Kanade. Tracking adversarial targets. In *International Conference on Machine Learning*, pages 369–377, 2014.

[AYLS19] Yasin Abbasi-Yadkori, Nevena Lazic, and Csaba Szepesvári. Model-free linear quadratic control via reduction to expert prediction. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3108–3117, 2019.

[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.

[B+15] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.

[BB08] Tamer Başar and Pierre Bernhard. *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer Science & Business Media, 2008.

[BCHM20] Nataly Brukhim, Xinyi Chen, Elad Hazan, and Shay Moran. Online agnostic boosting via regret minimization, 2020.

[BCP+16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[Bec17] Amir Beck. *First-order methods in optimization*. SIAM, 2017.

[BEP+18] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. In *arXiv:1808.04355*, 2018.

[Ber05] Dimitri Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 2005.

[BESK18] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

[BFH+18] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs. 2018.

[BH20] Nataly Brukhim and Elad Hazan. Online boosting with bandit feedback. *arXiv preprint arXiv:2007.11975*, 2020.

[Bha13] Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

[BHKL15] Alina Beygelzimer, Elad Hazan, Satyen Kale, and Haipeng Luo. Online gradient boosting. In *Advances in neural information processing systems*, pages 2458–2466, 2015.

[BHS21] Nataly Brukhim, Elad Hazan, and Karan Singh. A boosting approach to reinforcement learning. *arXiv preprint arXiv:2108.09767*, 2021.

[BKL15] Alina Beygelzimer, Satyen Kale, and Haipeng Luo. Optimal and adaptive algorithms for online boosting. In *International Conference on Machine Learning*, pages 2323–2331, 2015.

[BKT19] Maria-Florina Balcan, Mikhail Khodak, and Ameet Talwalkar. Provable guarantees for gradient-based meta-learning. In *International Conference on Machine Learning*, pages 424–433. PMLR, 2019.

[BM99] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In *Robustness in identification and control*, pages 207–226. Springer, 1999.

[BSO+16] Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1471–1479, 2016.

[CBFH+97] Nicolo Cesa-Bianchi, Yoav Freund, David Haussler, David P Helmbold, Robert E Schapire, and Manfred K Warmuth. How to use expert advice. *Journal of the ACM (JACM)*, 44(3):427–485, 1997.

[CBL06] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[CBS05] Nuttapong Chentanez, Andrew G. Barto, and Satinder P. Singh. Intrinsically motivated reinforcement learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1281–1288. MIT Press, 2005.

[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 1028–1037, 2018.

[CKM19] Alon Cohen, Tomer Koren, and Yishay Mansour. Learning linear-quadratic regulators efficiently with only $\sqrt{(}t)$ regret. *arXiv preprint arXiv:1902.06223*, 2019.

[CLL12] Shang-Tse Chen, Hsuan-Tien Lin, and Chi-Jen Lu. An online boosting algorithm with theoretical justifications. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1873–1880, 2012.

[CLL14] Shang-Tse Chen, Hsuan-Tien Lin, and Chi-Jen Lu. Boosting with online binary learners for the multiclass bandit problem. In *International Conference on Machine Learning*, pages 342–350, 2014.

[DB15] Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2818–2826, 2015.

[DFVR03] Daniela Pucci De Farias and Benjamin Van Roy. The linear programming approach to approximate dynamic programming. *Operations research*, 51(6):850–865, 2003.

[DH13] Ofer Dekel and Elad Hazan. Better rates for any adversarial deterministic MDP. In *International Conference on Machine Learning*, pages 675–683, 2013.

[DHK+11] Miroslav Dudik, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal learning for contextual bandits. *arXiv preprint arXiv:1106.2369*, 2011.

[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.

[dR96] Dick de Roover. Synthesis of a robust iterative learning controller using an h/sub/spl infin//approach. In *Proceedings of 35th IEEE Conference on Decision and Control*, volume 3, pages 3044–3049. IEEE, 1996.

[EDKM09] Eyal Even-Dar, Sham M Kakade, and Yishay Mansour. Online Markov decision processes. *Mathematics of Operations Research*, 34(3):726–736, 2009.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[FCRL17] Justin Fu, John Co-Reyes, and Sergey Levine. Ex2: Exploration with exemplar models for deep reinforcement learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach,

R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 2577–2587. Curran Associates, Inc., 2017.

[FGKM18] Maryam Fazel, Rong Ge, Sham M Kakade, and Mehran Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International Conference on Machine Learning*, pages 1466–1475, 2018.

[Fri02] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.

[FTM18] Mohamad Kazem Shirani Faradonbeh, Ambuj Tewari, and George Michailidis. Finite time identification in unstable linear systems. *Automatica*, 96:342–353, 2018.

[FW56] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics (NRL)*, 3(1-2):95–110, 1956.

[GHS+21] Paula Gradu, John Hallman, Daniel Suo, Alex Yu, Naman Agarwal, Udaya Ghai, Karan Singh, Cyril Zhang, Anirudha Majumdar, and Elad Hazan. Deluca–a differentiable control library: Environments, methods, and benchmarking. *arXiv preprint arXiv:2102.09968*, 2021.

[GKM18] Surbhi Goel, Adam Klivans, and Raghu Meka. Learning one convolutional layer with overlapping patches. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1783–1791, 2018.

[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.

[H+16] Elad Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[Haz16] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

[Haz19] Elad Hazan. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

[HCC+16] Rein Houthooft, Xi Chen, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. In D. D. Lee,

M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1109–1117. Curran Associates, Inc., 2016.

[HHLL18] Elad Hazan, Wei Hu, Yuanzhi Li, and Zhiyuan Li. Online improper learning with an approximation oracle. In *Advances in Neural Information Processing Systems*, pages 5652–5660, 2018.

[HK16] Elad Hazan and Tomer Koren. The computational power of optimization in online learning. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 128–141, 2016.

[HKS20] Elad Hazan, Sham Kakade, and Karan Singh. The nonstochastic control problem. In Aryeh Kontorovich and Gergely Neu, editors, *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, volume 117 of *Proceedings of Machine Learning Research*, pages 408–421, San Diego, California, USA, 08 Feb–11 Feb 2020. PMLR.

[HKSVS19] Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pages 2681–2691. PMLR, 2019.

[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.

[HMR18] Moritz Hardt, Tengyu Ma, and Benjamin Recht. Gradient descent learns linear dynamical systems. *The Journal of Machine Learning Research*, 19(1):1025–1068, 2018.

[HS21] Elad Hazan and Karan Singh. Boosting for online convex optimization. *arXiv preprint arXiv:2102.09305*, 2021.

[HSZ17a] Elad Hazan, Karan Singh, and Cyril Zhang. Efficient regret minimization in non-convex games. In *International Conference on Machine Learning*, pages 1433–1441. PMLR, 2017.

[HSZ17b] 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.

[HSZ18] Elad Hazan, Karan Singh, and Cyril Zhang. Dynamic learning system, December 13 2018. US Patent App. 15/914,967.

[HWMZ20] Lukas Hewing, Kim P Wabersich, Marcel Menner, and Melanie N Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296, 2020.

[HZAL18] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018.

[JGT17] Young Hun Jung, Jack Goetz, and Ambuj Tewari. Online multiclass boosting. In *Advances in neural information processing systems*, pages 919–928, 2017.

[JT18] Young Hun Jung and Ambuj Tewari. Online boosting algorithms for multi-label ranking. In *International Conference on Artificial Intelligence and Statistics*, pages 279–287, 2018.

[Kae93] Leslie Pack Kaelbling. Learning to achieve goals. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, Chambery, France, 1993. Morgan Kaufmann.

[Kak03] Sham Machandranath Kakade. *On the sample complexity of reinforcement learning.* PhD thesis, University of London London, England, 2003.

[Kal60] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82.1:35–45, 1960.

[KKL09] Sham M Kakade, Adam Tauman Kalai, and Katrina Ligett. Playing games with approximation algorithms. *SIAM Journal on Computing*, 39(3):1088–1106, 2009.

[Kry08] Nikolaj Vladimirovič Krylov. *Controlled diffusion processes*, volume 14. Springer Science & Business Media, 2008.

[KS02] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

[LA12] Shiau Hong Lim and Peter Auer. Autonomous exploration for navigating in mdps. In *Conference on Learning Theory*, pages 40–1, 2012.

[LCRM04] Wilbur Langson, Ioannis Chryssochoos, SV Raković, and David Q Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125–133, 2004.

[LH14] Tor Lattimore and Marcus Hutter. Near-optimal pac bounds for discounted mdps. *Theoretical Computer Science*, 558:125–143, 2014.

[Lju98] Lennart Ljung. *System identification: Theory for the User*. Prentice Hall, Upper Saddle Riiver, NJ, 2 edition, 1998.

[LLTyO12] Manuel Lopes, Tobias Lang, Marc Toussaint, and Pierre yves Oudeyer. Exploration in model-based reinforcement learning by empirically estimating learning progress. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 206–214. Curran Associates, Inc., 2012.

[LSRB09] Christian Leistner, Amir Saffari, Peter M Roth, and Horst Bischof. On robustness of on-line boosting-a competitive study. In *IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 1362–1369. IEEE, 2009.

[LT04] Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)*, pages 222–229, 2004.

[LW94] N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212 – 261, 1994.

[LZ08] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, pages 817–824, 2008.

[May14] David Q Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

[Mit97] Tom Mitchell. Machine learning. 1997.

[MJR15] Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2125–2133. Curran Associates, Inc., 2015.

[MKS+15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[Moo12] Kevin L Moore. *Iterative learning control for deterministic systems*. Springer Science & Business Media, 2012.

[MSR05] David Q Mayne, María M Seron, and SV Raković. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41(2):219–224, 2005.

[MTR19] Horia Mania, Stephen Tu, and Benjamin Recht. Certainty equivalent control of lqr is efficient. *arXiv preprint arXiv:1902.07826*, 2019.

[NHR99] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, 1999.

[NJG17] Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

[NK05] Andrew Y Ng and HJ Kim. Stable adaptive control with online learning. In *Advances in Neural Information Processing Systems*, pages 977–984, 2005.

[NPD+18] Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *CoRR*, abs/1807.04742, 2018.

[NS04] Jun Nakanishi and Stefan Schaal. Feedback error learning and nonlinear adaptive control. *Neural Networks*, 17(10):1453–1465, 2004.

[OAC18] Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8625–8637. Curran Associates, Inc., 2018.

[OBPVR16] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In D. D. Lee, M. Sugiyama, U. V. Luxburg,

I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4026–4034. Curran Associates, Inc., 2016.

[OBvdOM17] Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2721–2730, 2017.

[OH05] David H Owens and Jari Hätönen. Iterative learning control—an optimization paradigm. *Annual reviews in control*, 29(1):57–70, 2005.

[Ols01] Richard Olshen. A conversation with Leo Breiman. *Statistical Science*, 16(2):184–198, 2001.

[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.

[PAED17] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

[PBGM62] Lev S Pontryagin, VG Boltyanskii, RV Gamkrelidze, and EF Mishchenko. The mathematical theory of optimal processes, translated by kn trirogoff. *New York*, 1962.

[PMA10] Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[Put14] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[RB17] Ugo Rosolia and Francesco Borrelli. Learning model predictive control for iterative tasks. a data-driven control framework. *IEEE Transactions on Automatic Control*, 63(7):1883–1896, 2017.

[RGB11] Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.

[Ros15] I Michael Ross. *A primer on Pontryagin's principle in optimal control*. Collegiate publishers, 2015.

[RS16] Alexander Rakhlin and Karthik Sridharan. Bistro: An efficient relaxation-based method for contextual bandits. In *ICML*, pages 1977–1985, 2016.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[SBR19] Max Simchowitz, Ross Boczar, and Benjamin Recht. Learning linear dynamical systems with semi-parametric least squares. *arXiv preprint arXiv:1902.00768*, 2019.

[SF12a] Robert E Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. MIT Press, 2012.

[SF12b] Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. Cambridge university press, 2012.

[SHM+16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[Sim20] Max Simchowitz. Making non-stochastic control (almost) as easy as stochastic. *arXiv preprint arXiv:2006.05910*, 2020.

[SKS16] Vasilis Syrgkanis, Akshay Krishnamurthy, and Robert Schapire. Efficient algorithms for adversarial contextual learning. In *International Conference on Machine Learning*, pages 2159–2168, 2016.

[SLB09] Satinder Singh, Richard L. Lewis, and Andrew G. Barto. Where do rewards come from? *Proceedings of the Annual Conference of the Cognitive Science Society (CogSci)*, 2009.

[SLBS10] Satinder P. Singh, Richard L. Lewis, Andrew G. Barto, and Jonathan Sorg. Intrinsically motivated reinforcement learning: An evolutionary perspective. *IEEE Transactions on Autonomous Mental Development*, 2:70–82, 2010.

[SLKS16] Vasilis Syrgkanis, Haipeng Luo, Akshay Krishnamurthy, and Robert E Schapire. Improved regret bounds for oracle-based adversarial contextual bandits. In *Advances in Neural Information Processing Systems*, pages 3135–3143, 2016.

[SLW+06] Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888. ACM, 2006.

[SMSM00] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[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. *arXiv preprint arXiv:1802.08334*, 2018.

[SR19] Tuhin Sarkar and Alexander Rakhlin. Near optimal finite time identification of arbitrary linear dynamical systems. In *International Conference on Machine Learning*, pages 5610–5618, 2019.

[SRD19] Tuhin Sarkar, Alexander Rakhlin, and Munther A Dahleh. Finite-time system identification for partially observed lti systems of unknown order. *arXiv preprint arXiv:1902.01848*, 2019.

[SRM+18] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, Marc Pollefeys, Timothy P. Lillicrap, and Sylvain Gelly. Episodic curiosity through reachability. *CoRR*, abs/1810.02274, 2018.

[SS10] István Szita and Csaba Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1031–1038, 2010.

[SS+12] Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[SSH20] Max Simchowitz, Karan Singh, and Elad Hazan. Improper learning for non-stochastic control. In *Conference on Learning Theory*, pages 3320–3436. PMLR, 2020.

[Ste94a] Robert F Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

[Ste94b] Robert F Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

[Str] Gilbert Strang. *Introduction to linear algebra*, volume 3.

[SZG+21] Daniel Suo, Cyril Zhang, Paula Gradu, Udaya Ghai, Xinyi Chen, Edgar Minasyan, Naman Agarwal, Karan Singh, Julienne LaChance, Tom Zajdel, et al. Machine learning for mechanical ventilation control. *arXiv preprint arXiv:2102.06779*, 2021.

[Ted20] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832)*. 2020.

[THF+17] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 2753–2762. Curran Associates, Inc., 2017.

[TL05] Emanuel Todorov and 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.

[Tod07] Emanuel Todorov. Linearly-solvable markov decision problems. In *Advances in neural information processing systems*, pages 1369–1376, 2007.

[Tu19] Stephen Lyle Tu. *Sample Complexity Bounds for the Linear Quadratic Regulator*. PhD thesis, UC Berkeley, 2019.

[Vap99] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

[VJ01] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[WCSB19] Nolan Wagener, Ching-An Cheng, Jacob Sacks, and Byron Boots. An online learning approach to model predictive control. *arXiv preprint arXiv:1902.08967*, 2019.

[WdWK+18] David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *CoRR*, abs/1811.11359, 2018.

[WMD19] Yuh-Shyang Wang, Nikolai Matni, and John C Doyle. A system level approach to controller synthesis. *IEEE Transactions on Automatic Control*, 2019.

[WRR+17] Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomenech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*, 2017.

[YMS09] Jia Yuan Yu, Shie Mannor, and Nahum Shimkin. Markov decision processes with arbitrary reward processes. *Mathematics of Operations Research*, 34(3):737–757, 2009.

[ZD98] Kemin Zhou and John Comstock Doyle. *Essentials of robust control*, volume 104. Prentice hall Upper Saddle River, NJ, 1998.

[ZDG96a] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice-Hall, Inc., USA, 1996.

[ZDG+96b] Kemin Zhou, John Comstock Doyle, Keith Glover, et al. *Robust and optimal control*, volume 40. Prentice hall New Jersey, 1996.

[Zin03] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.

[ZOS18] Zeyu Zheng, Junhyuk Oh, and Satinder Singh. On learning intrinsic rewards for policy gradient methods. *CoRR*, abs/1804.06459, 2018.