Title
Reinforcement Learning in Structured and Partially Observable Environments

Author
Azizzadenesheli, Kamyar

Publication Date
2019

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,
IRVINE

Reinforcement Learning in Structured and Partially Observable Environments

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Electrical Engineering and Computer Science

by

Kamyar Azizzadenesheli

Dissertation Committee:
Professor Sameer Singh, Chair
Professor Marco Levorato
Professor Animashree Anandkumar

2019

DEDICATION

To my parents without whom this journey wouldn't be taken.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Page

List of Algorithms

ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Anima Anandkumar for the continuous support of my Ph.D. study and related research, for her patience, motivation, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D. study. I would like to express my most sincere gratitude to Prof. Yisong Yue for unbelievable support during my stay at Caltech and before. For all the advice and sharing of the most valuable experiences of his which saved my career multiple times.

My sincere thanks go to my second advisor Professor Sameer Singh for all the supports during my Ph.D. career and all the wise advice he gave me to through my crucial decisions along my Ph.D. journey.

Besides my advisors, I would like to thank Professor Marco Levorato, my thesis committee member and a great person whom I had a chance to know, for his insightful comments, guidance, and encouragement during my Ph.D. career.

I thank my fellow lab-mates and friends for all the fun we have had in the last few years. I particularly thank Dr. Forough Arabshahi for her kindness and advice. She has been always available to help me through my career. I would like to thank Prof. Furong Huang, Dr. Hanie Sedghi, Dr. Majid Janzamin, Dr. Anqi(Angie) Liu, Dr. Yang Shi, Jeremy Bernstein, Sahin Lale for enlightening me in conducting proper research.

I would like to thank my parents, my brother, his beloved fiancee, and relatives for supporting me spiritually throughout writing this thesis and my life in general.

Last but not least, I would also thank all my friends who have been by my side all these years, and without whom this journey would not have been possible.

During my Ph.D. career, I was honored to know many great people who mainly are now great friends of mine. I would like to sincerely thank Professor Csaba Szepesvari for carving my thoughts and giving me advice without any hesitation. I would like to thank Professor Zachary Chase Lipton who plaid a significant role in my academic and personal life. I thank all my colleagues at INRIA, particularly Dr. Alessandro Lazaric whose amazing helps and supports made my first steps in my research career. There is no word to appreciate his priceless patient effort in helping me. I thank my colleagues at Microsoft Research lab for all the fantastic discussion which enlightened my research path.

I would like to thank my colleagues at UC Berkeley at Simons Institute and participants in Foundation Machine Learning program, in particular, Prof. Daniel Hsu and Prof. Peter Bartlett for their insights in research and teaching me fundamental ways of thinking. I would like to thank my colleagues at Stanford University, especially my sincere gratitude to my host, Prof. Emma Brunskill for all the valuable discussions and lessons. I also appreciate the exceptional support and kindness by my colleagues and friends at Stanford University, Ramtin Keramati, Khasahyar Khosravi, and Behrad Habib Afshar. I thank my amazing

colleagues and friends at Caltech, who showed me a new aspect of academic and personal life. I thank them for all the lessons they taught me. Finally, I appreciate all the people who helped me to deliver proper research and due to the space limitation could not bring their name on this draft.

My sincere thanks to my grant provider and funding agencies, NSF, Army Research, Air Force, Office of Naval Research.

CURRICULUM VITAE

Kamyar Azizzadenesheli

EDUCATION

Doctor of Philosophy in Electrical Engineering & Computer Science **2019**
University of California, Irvine, *Irvine, CA.*

Master of Science in Electrical Engineering & Computer Science **2015**
University of California, Irvine, *Irvine, CA.*

Bachelor of Science in Computational Sciences **2007**
Sharif University of Technology *Tehran, Iran*

RESEARCH EXPERIENCE

Special Student **Spring 2019**
California Institute of Technology *Pasadena, CA.*

Visiting Student Researcher **Summer 2018–Spring 2019**
California Institute of Technology *Pasadena, CA.*

Visiting Student Researcher **Fall 2017–Summer2018**
Stanford University *Stanford, CA.*

Long-term Visiting Researcher **Spring 2017–Summer2017**
Simons Institute, University of California, Berkeley *Berkeley, CA.*

Guest Researcher **Summer 2016**
INRIA *Lille, France*

Visiting researcher **Summer 2016**
Microsoft Research Lab *Boston, MA.*

Visiting researcher **Summer 2016**
Microsoft Research Lab *New York, MA.*

Graduate Research Assistant **Summer 2014–Summer 2019**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant University of California, Irvine	**Winder 2015** *Irvine, CA.*
Teaching Assistant California Institute of Technology	**Winder 2019** *Pasadena, CA.*
Teaching Assistant Advanced High Schools	**2009–2014** *Iran*

Books

- **Deep Learning - The Straight Dope**, an online Deep Learning book on Amazon Mxnet Library. Zachary C. Lipton, Mu Li, Alex Smola, Sheng Zha, Aston Zhang, Joshua Z. Zhang, Eric Junyuan Xie, **K. Azizzadenesheli**, Jean Kossaifi, Stephan Rabanser, `[link]`

Publication

1. **K. Azizzadenesheli**, Manish Kumar Bera, Animashree Anandkumar. Trust Region Policy Optimization of POMDPs, `[paper]`

2. Sahin Lale, **K. Azizzadenesheli**, Babak Hassibi, Animashree Anandkumar. Stochastic Linear Bandits with Hidden Low Rank Structure, `[paper]`

3. **K. Azizzadenesheli**, Anqi Liu, Fanny Yang, Animashree Anandkumar. Regularized Learning for Domain Adaptation under Label Shifts, `[paper]`
   Appeared at International Conference on Learning Representations (ICLR) 2019

4. Jeremy Bernstein, Jiawei Zhao, **K. Azizzadenesheli**, Anima Anandkumar. signSGD with Majority Vote is Communication Efficient and Fault Tolerant, `[paper]`
   Appeared at International Conference on Learning Representations (ICLR) 2019

5. Guanya Shi, Xichen Shi, Michael O'Connell1, Rose Yu, **K. Azizzadenesheli**, Animashree Anandkumar, Yisong Yue, and Soon-Jo Chung. Neural Lander: Stable Drone Landing Control using Learned Dynamics, `[paper]` `[video]`
   International Conference on Robotics and Automation (ICRA) 2019

6. **K. Azizzadenesheli**, Brandon Yang, Weitang Liu, Emma Brunskill, Zachary C Lipton, Animashree Anandkumar. Surprising Negative Results for Generative Adversarial Tree Search, `[paper]`
   Appeared at International Conference on Machine Learning (ICML) 2018 workshop

7. Jeremy Bernstein, Yu-Xiang Wang, **K. Azizzadenesheli**, Anima Anandkumar. signSGD: Compressed Optimisation for Non-Convex Problems, `[paper]`
   Appeared at International Conference on Machine Learning (ICML) 2018

8. Jeremy Bernstein, **K. Azizzadenesheli**, Yu-Xiang Wang, Anima Anandkumar. Compression by the signs: distributed learning is a two-way street, `[paper]`
   Appeared at International Conference on Learning Representations (ICLR) 2018 Workshop

9. Guneet S. Dhillon, **K. Azizzadenesheli**, Jeremy D. Bernstein, Jean Kossaifi, Aran Khanna, Zachary C. Lipton, Animashree Anandkumar. Stochastic activation pruning for robust adversarial defense, [paper]
Appeared at International Conference on Learning Representations (ICLR) 2017

10. **K. Azizzadenesheli**, Animashree Anandkumar. Efficient Exploration through Bayesian Deep Q-Networks, *Appeared at Neural Information Processing Systems,* [paper] [talk]
Appeared at Neural Information Processing Systems (NeurIPS) 2017 Workshop

11. **K. Azizzadenesheli**, Alessandro Lazaric, Anima Anandkumar. Reinforcement Learning in Rich Observation MDPs using Spectral Methods, [paper]
Appeared at Multi-disciplinary Conference on Reinforcement Learning and Decision Making (RLDM) 2017

12. Zachary C. Lipton, **K. Azizzadenesheli**, Abhishek Kumar, Lihong Li, Jianfeng Gao, Li Deng. Combating Reinforcement Learning's Sisyphean Curse with Intrinsic Fear, [paper]
Appeared at Neural Information Processing Systems (NeurIPS) 2016 Workshop

13. **K. Azizzadenesheli**, Alessandro Lazaric, Anima Anandkumar. Experimental paper: Reinforcement Learning of POMDPs using Spectral Methods, [paper]
Appeared at Neural Information Processing Systems (NeurIPS) 2016 Workshop

14. **K. Azizzadenesheli**, Alessandro Lazaric, Anima Anandkumar. Open Problem: Approximate Planning of POMDPs in the class of Memoryless Policies, [paper] [talk]
Appeared at Conference on Learning Theory (COLT) 2016

15. **K. Azizzadenesheli**, Alessandro Lazaric, Anima Anandkumar. Reinforcement Learning of POMDPs using Spectral Methods, [paper] [talk]
Appeared at Conference on Learning Theory (COLT) 2016

Azizzadenesheli et al. (2016a), Azizzadenesheli et al. (2016c), Azizzadenesheli et al. (2016b), Lipton et al. (2016a), Azizzadenesheli et al. (2017), Azizzadenesheli and Anandkumar (2018), Bernstein et al. (2018a), Bernstein et al. (2018b), Dhillon et al. (2018), Azizzadenesheli et al. (2018b), Azizzadenesheli et al. (2019), Bernstein et al. (2018c), Azizzadenesheli et al. (2018a), Shi et al. (2018), Lale et al. (2019), Azizzadenesheli (2019).

ABSTRACT OF THE DISSERTATION

Reinforcement Learning in Structured and Partially Observable Environments

By

Kamyar Azizzadenesheli

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California, Irvine, 2019

Professor Sameer Singh, Chair

Sequentially making-decision abounds in real-world problems ranging from robots needing to interact with humans to companies aiming to provide reasonable services to their customers. It is as diverse as self-driving cars, health-care, agriculture, robotics, manufacturing, drug discovery, and aerospace. Reinforcement Learning (RL), as the study of sequential decision-making under uncertainty, represents a core aspect challenges in real-world applications.

While most of the practical application of interests in RL are high dimensions, we study RL problems from theory to practice in high dimensional, structured, and partially observable settings. We show how statistically develop efficient RL algorithm for a variety of RL problems, from recommendation systems to robotics and games. We theoretically study these problems from their first principles to provide RL agents which efficiently interact with their surrounding environment and learn the desired behavior while minimizing their regrets.

We study linear bandit problems where we propose Projected Stochastic Linear Bandit (PSLB), upper confidence bound based algorithm in linear bandit which exploit the intrinsic structure of the decision-making problem to significantly enhance the performance of RL agents.

We study the problem of RL in Markov Decision Process (MDP) where we propose the

first sample efficient model-free algorithm for the general continuous state and action space MDPs. We further investigate safe RL setting and introduce a safe RL algorithm to avoid catastrophic mistakes that can be made by an RL agent. We extensively study tree-based methods, a well-popularized method in RL which is also the core to Alpha-Go, a technique to beat the masters of board games such as Go game.

We extend our study to partially observable environments, such as partially observable Markov decision processes (POMDP) where we propose the first regret analysis for the class of memoryless policies. We continue this study to a class of problems known as rich observable Markov decision processes (ROMPD) and propose the first regret bound with no dependency in the ambient dimension in the dominating terms.

We empirically study the significance of all these theoretically guaranteed methods and show their value in practice.

Chapter 1

Introduction

1.1 Motivation

Reinforcement Learning (RL) is an effective approach to solve the problem of sequential decision–making under uncertainty. RL agents learn how to maximize long-term reward using the experience obtained by direct interaction with a stochastic environment (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998). Since the environment is initially unknown, the agent needs to balance between *exploring* the environment to estimate its structure, and *exploiting* the estimates to compute a policy that maximizes the long-term reward. As a result, designing a RL algorithm requires three different elements: **1)** an estimator for the environment's structure, **2)** a planning algorithm to compute the optimal policy of the estimated environment (LaValle, 2006), and **3)** a strategy to make a trade-off between exploration and exploitation to minimize the *regret*, i.e., the difference between the performance of the exact optimal policy and the rewards accumulated by the agent over time.

While most of the practical application of interests in RL are high dimensions, we study RL problems from theory to practice in high dimensional, structured, and partially observ-

able settings. We show how statistically develop efficient RL algorithm for a variety of RL environment, from recommendation systems to robotics and games. We theoretically study these problems from their first principles to provide RL agents which efficiently interact with their surrounding environment and learn the desired behavior while minimizing regret.

1.2 Summery of Contribution

We study linear bandit problems where we propose Projected Stochastic Linear Bandit (PSLB), upper confidence bound based algorithm in linear bandit which exploit the intrinsic structure of the decision-making problem to enhance the performance of RL agent significantly.

High-dimensional representations often have a lower dimensional underlying structure. This is particularly the case in many decision making settings. For example, when the representation of actions is generated from a deep neural network, it is reasonable to expect a low-rank structure whereas conventional structures like sparsity are not valid anymore. Subspace recovery methods, such as Principle Component Analysis (PCA) can find the underlying low-rank structures in the feature space and reduce the complexity of the learning tasks. In this work, we propose Projected Stochastic Linear Bandit (PSLB), an algorithm for high dimensional stochastic linear bandits (SLB) when the representation of actions has an underlying low-dimensional subspace structure. PSLB deploys PCA based projection to iteratively find the low rank structure in SLBs. We show that deploying projection methods assures dimensionality reduction and results in a tighter regret upper bound that is in terms of the dimensionality of the subspace and its properties, rather than the dimensionality of the ambient space. We modify the image classification task into the SLB setting and empirically show that, when a pre-trained DNN provides the high dimensional feature representations, deploying PSLB results in significant reduction of regret and faster convergence

to an accurate model compared to state-of-art algorithm (Lale et al., 2019).

We study the problem of RL in Markov Decision Process (MDP) where we propose the first sample efficient model-free algorithm for the general continuous state and action space MDPs.

We study reinforcement learning (RL) in high dimensional episodic Markov decision processes (MDP). We consider value-based RL when the optimal Q-value is a linear function of d-dimensional state-action feature representation. For instance, in deep-Q networks (DQN), the Q-value is a linear function of the feature representation layer (output layer). We propose two algorithms, one based on optimism, LinUCB, and another based on posterior sampling, LinPSRL. We guarantee frequentist and Bayesian regret upper bounds of $\widetilde{\mathcal{O}}(d\sqrt{T})$ for these two algorithms, where T is the number of episodes. We extend these methods to deep RL and propose Bayesian deep Q-networks (BDQN), which uses an efficient Thompson sampling algorithm for high dimensional RL. We deploy the double DQN (DDQN) approach, and instead of learning the last layer of Q-network using linear regression, we use Bayesian linear regression, resulting in an approximated posterior over Q-function. This allows us to directly incorporate the uncertainty over the Q-function and deploy Thompson sampling on the learned posterior distribution resulting in efficient exploration/exploitation trade-off. We empirically study the behavior of BDQN on a wide range of Atari games. Since BDQN carries out more efficient exploration and exploitation, it is able to reach higher return substantially faster compared to DDQN (Azizzadenesheli and Anandkumar, 2018).

We further investigate safe RL setting and introduce a safe RL algorithm to avoid catastrophic mistakes that can be made by an RL agent.

Many practical environments contain catastrophic states that an optimal agent would visit infrequently or never. Even on toy problems, Deep Reinforcement Learning (DRL) agents tend to periodically revisit these states upon forgetting their existence under a new policy.

We introduce *intrinsic fear* (IF), a learned reward shaping that guards DRL agents against periodic catastrophes. IF agents possess a *fear* model trained to predict the probability of imminent catastrophe. This score is then used to penalize the Q-learning objective. Our theoretical analysis bounds the reduction in average return due to learning on the perturbed objective. We also prove robustness to classification errors. As a bonus, IF models tend to learn faster, owing to reward shaping. Experiments demonstrate that *intrinsic-fear* DQNs solve otherwise pathological environments and improve on several Atari games (Lipton et al., 2016a).

We extensively study tree-based methods, a well-popularized method in RL which is also the core to Alpha-Go, a technique to beat the masters of board games such as Go game.

While many recent advances in deep reinforcement learning rely on model-free methods, model-based approaches remain an alluring prospect for their potential to exploit unsupervised data to learn environment dynamics. One prospect is to pursue hybrid approaches, as in AlphaGo, which combines Monte-Carlo Tree Search (MCTS)—a model-based method—with deep-Q networks (DQNs)—a model-free method. MCTS requires generating rollouts, which is computationally expensive. In this paper, we propose to simulate roll-outs, exploiting the latest breakthroughs in image-to-image transduction, namely Pix2Pix GANs, to predict the dynamics of the environment. Our proposed algorithm, *generative adversarial tree search* (GATS), simulates rollouts up to a specified depth using both a GAN-based dynamics model and a reward predictor. GATS employs MCTS for planning over the simulated samples and uses DQN to estimate the Q-function at the leaf states. Our theoretical analysis establishes some favorable properties of GATS vis-a-vis the bias-variance trade-off and empirical results show that on 5 popular Atari games, the dynamics and reward predictors converge quickly to accurate solutions. However, GATS fails to outperform DQNs. Notably, in these experiments, MCTS has only short rollouts (up to tree depth 4), while previous successes of MCTS have involved tree depth in the hundreds. We present a hypothesis for why tree search with

short rollouts can fail even given perfect modeling (Azizzadenesheli et al., 2018b).

We extend our study to partially observable environments, such as partially observable Markov decision processes (POMDP) where we propose the first regret analysis for the class of memoryless policies. We extend this study to a class of problems known as rich observable Markov decision processes (ROMPD) and proposed the first regret bound with no dependency in the ambient dimension in the dominating term.

We propose a new reinforcement learning algorithm for partially observable Markov decision processes (POMDP) based on spectral decomposition methods. While spectral methods have been previously employed for consistent learning of (passive) latent variable models such as hidden Markov models, POMDPs are more challenging since the learner interacts with the environment and possibly changes the future observations in the process. We devise a learning algorithm running through episodes, in each episode we employ spectral techniques to learn the POMDP parameters from a trajectory generated by a fixed policy. At the end of the episode, an optimization oracle returns the optimal memoryless planning policy which maximizes the expected reward based on the estimated POMDP model. We prove an order-optimal regret bound w.r.t. the optimal memoryless policy and efficient scaling with respect to the dimensionality of observation and action spaces (Azizzadenesheli et al., 2016c,a,a).

Reinforcement learning (RL) in Markov decision processes (MDPs) with large state spaces is a challenging problem. The performance of standard RL algorithms degrades drastically with the dimensionality of state space. However, in practice, these large MDPs typically incorporate a latent or hidden low-dimensional structure. In this paper, we study the setting of *rich-observation* Markov decision processes (ROMDP), where there are a small number of hidden states which possess an injective mapping to the observation states. In other words, every observation state is generated through a single hidden state, and this mapping is unknown a priori. We introduce a spectral decomposition method that consistently learns this mapping, and more importantly, achieves it with low regret. The estimated mapping is

integrated into an optimistic RL algorithm (UCRL), which operates on the estimated hidden space. We derive finite-time regret bounds for our algorithm with a weak dependence on the dimensionality of the observed space. In fact, our algorithm asymptotically achieves the same average regret as the oracle UCRL algorithm, which has the knowledge of the mapping from hidden to observed spaces. Thus, we derive an efficient spectral RL algorithm for ROMDPs (Azizzadenesheli et al., 2016b).

We propose Generalized Trust Region Policy Optimization (GTRPO), a policy gradient Reinforcement Learning (RL) algorithm for both Markov decision processes (MDP) and Partially Observable Markov Decision Processes (POMDP). Policy gradient is a class of model-free RL methods. Previous policy gradient methods are guaranteed to converge only when the underlying model is an MDP and the policy is run for an infinite horizon. We relax these assumptions to episodic settings and to partially observable models with memoryless policies. For the latter class, GTRPO uses a variant of the Q-function with only three consecutive observations for each policy updates, and hence, is computationally efficient. We theoretically show that the policy updates in GTRPO monotonically improve the expected cumulative return and hence, GTRPO has convergence guarantees (Azizzadenesheli et al., 2018a).

We empirically study the significance of all these theoretically guaranteed methods and show their importance in practice.

We conclude this section with a reference to our study on ways to extend the state of RL to practical and high dimensional settings (Azizzadenesheli, 2019).

1.3 Background

In RL, we study the interaction between an agent and an environment. At each time step t the agent receives an observation from the environment and then act accordingly while the agent does not have a clear understanding of its environment. In order to build a better intuition on RL, imagine a newborn baby. In the beginning, she does not have a clear understanding of her surrounding environment. The baby interacts with her environment to build knowledge and act accordingly. If the baby's behavior is good, or the action taken by her is useful, the baby receives a reward in some notion in any means the reader would like to think of. It can be candy, treat, or even internal Dopamine released in her brain. These rewards and interactions help the baby to build a better understanding of how to behave in this world.

In RL, at each time step, the environment is at some state, and the agent receives some observation from the environment. The state in general case is hidden from the agent. This observation can be the state itself, like in many simple video games, or it can be a sensory, noisy, and partial observation of the state. After receiving the observation, the agent makes its decision, called the action. As a result of this decision, the agent receives some reward as feedback and the environment evolves to a new hidden state.

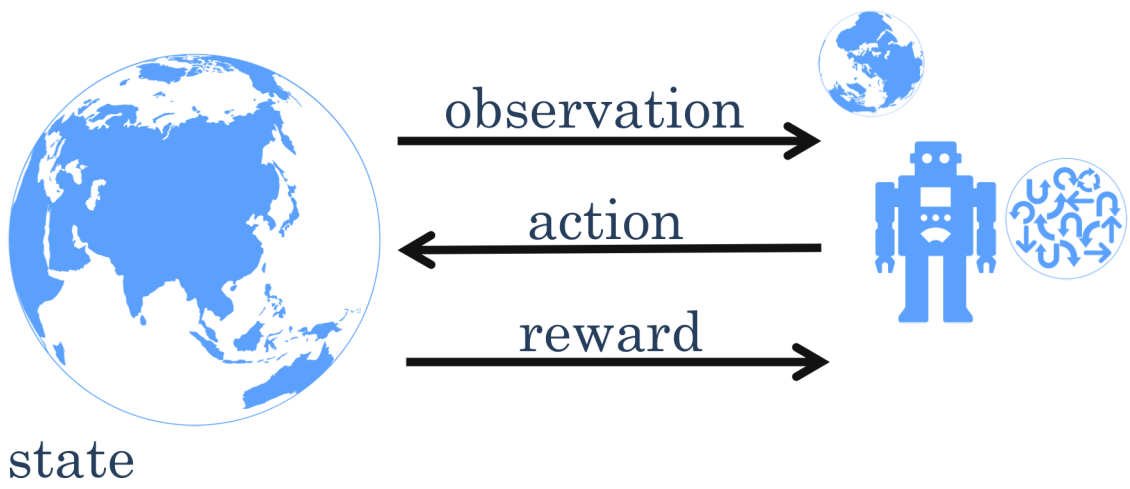


Figure 1.1: RL paradigm

Chapter 2

RL in Linear Bandits

Stochastic Linear Bandits with Hidden Low Rank Structure

We propose Projected Stochastic Linear Bandits (PSLB), a sequential decision-making algorithm for high dimensional stochastic linear bandits (SLB). We show that when the representations of actions inherit an unknown low-dimensional subspace structure, PSLB deploys subspace recovery methods, e.g., principal component analysis, and efficiently recovers this structure. PSLB exploits this structure to better guide the exploration and exploitation, resulting in significant improvement in performance. We prove that PSLB notably advances the previously known regret upper bound and obtains a regret upper bound which scales with the intrinsic dimension of the subspace, rather than the large ambient-dimension of the action space. We empirically study PSLB on a range of image classification tasks formulated as bandit problems. We show that, when a pre-trained DNN provides the high dimensional action (label) representations, deploying PSLB results in a significant reduction in the regret and faster convergence to an accurate model compared to the state-of-art algorithm.

2.1 Introduction

Stochastic linear bandits (SLB) is a problem of sequential decision-making under uncertainty. At each round of SLB, an agent chooses an action from a decision set and receives a stochastic reward from the environment whose expected value is an unknown linear function of the d-dimensional action representation vector. The agent's goal is to maximize its cumulative reward. Thus, it dedicates the actions to not only maximize the current reward but also to explore other actions to build a better estimation of the unknown linear function and guarantee higher future rewards. Through the course of interactions, the agent implicitly or explicitly constructs the model of the environment in order to systematically balance the trade-off between *exploration* and *exploitation*.

The lack of knowledge of the true environment model causes the agent to make mistakes by picking sub-optimal actions. The agent aims to design a strategy to minimize the cumulative cost of these mistakes, known as regret. One promising approach is to utilize the *optimism in the face of uncertainty* (OFU) principle (Lai and Robbins, 1985). OFU based methods estimate the environment model up to a confidence interval and construct a set of plausible models within this interval. Among those models, they choose the most optimistic one and follow the optimal behavior suggested by the selected model.

For general SLB problems, Abbasi-Yadkori et al. (2011) deploys the OFU principle, proposes OFUL algorithm, and for a d-dimensional SLB, derives a regret upper bound of $\widetilde{\mathcal{O}}\left(d\sqrt{T}\right)$. These regret bounds in high dimensional problems especially when d and T are about the same order are not practically tolerable. Fortunately, real-world problems may contain hidden low-dimensional structures. For example in classical recommendation systems, each item is represented by a large and highly detailed hand-engineered feature vector; but not all the components of the features are helpful for the recommendation task. Therefore, the true underlying linear function in SLBs is highly sparse. Abbasi-Yadkori et al. (2012)

and Carpentier and Munos (2012) show how to exploit this additional structure and derive regret upper bound of $\widetilde{\mathcal{O}}\left(\sqrt{sdT}\right)$ and $\widetilde{\mathcal{O}}\left(s\sqrt{T}\right)$ respectively where s is the sparsity level of the true underlying linear function. The recent success of deep neural networks (DNN) in representation learning provide significant promises in advancing machine learning to high dimensional real-world tasks (LeCun et al., 1998). DNNs convolve the raw features of the input and construct new feature representations which arguably can replace the hand-engineered feature vectors. When a DNN provides the feature representations, the sparse structure is not relevant anymore; instead, the low-rank structure is suitable.

At each round of SLB, the chosen action is assigned a supervised reward signal while other actions in the decision sets remain unsupervised. Even though the primary motivation in the SLB framework is decision-making within a large stochastic decision set, the majority of prior works do not exploit possible hidden structures in these sets. For example, Abbasi-Yadkori et al. (2011) utilizes only the supervised actions, i.e., the actions selected by the algorithm, to construct the environment model. It ignores all other unsupervised actions in the decision set. On the contrary, large number of actions in the decision sets can be useful in reducing the dimension of the problem and simplifying the learning task.

Contributions: In this paper, we demonstrate a method to utilize unsupervised actions in the decision sets to improve the performance in SLB. We deploy subspace recovery using principle component analysis (PCA) to exploit the structure in the massive number of unsupervised actions observed in the decision sets of SLB and reduce the dimensionality and the complexity of SLBs. We propose PSLB, an algorithm for high dimensional SLB, and show that if actions come from a perturbed m-dimensional subspace, deploying PSLB improves the regret upper bound to $min\left\{\widetilde{\mathcal{O}}\left(\Upsilon\sqrt{T}\right), \widetilde{\mathcal{O}}\left(d\sqrt{T}\right)\right\}$. Here Υ captures the effect of difficulty of subspace recovery in SLB as a function of the structure of the problem. If the underlying subspace is easily identifiable, *e.g.*, large decision sets are available in each round, using subspace recovery provides faster learning of the underlying linear function;

thus, smaller regret. In contrast, if learning the subspace is hard, *e.g.*, the number of actions (unsupervised signals) in each round is limited, then subspace recovery based approaches do not provide much benefit in learning the underlying system.

We adapt the image classification tasks on MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky and Hinton, 2009) and ImageNet (Krizhevsky et al., 2012) datasets to the SLB framework and apply both PSLB and OFUL on these datasets. We observe that there exists a low dimensional subspace in the feature space when a pre-trained DNN produces the d-dimensional feature vectors. We empirically show that PSLB learns the underlying model significantly faster than OFUL and provides orders of magnitude smaller regret in SLBs obtained from MNIST, CIFAR-10, and ImageNet datasets.

2.2 Preliminaries

For any positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$. The Euclidean norm of a vector x is denoted by $\|x\|_2$. The spectral norm of matrix A is denoted by $\|A\|_2$. $A^\dagger$ denotes the Moore-Penrose inverse of matrix A. For any positive definite matrix M, $\|x\|_M$ denote the norm of a vector x defined as $\|x\|_M := \sqrt{x^T M x}$. The j-th eigenvalue of a symmetric matrix A is denoted by $\lambda_j(A)$, where $\lambda_1(A) \geq \lambda_2(A) \geq \ldots$. I_d denotes $d \times d$ identity matrix. If Y_i is a column vector then $\mathbf{Y}_t$ is a matrix whose columns are $Y_1, \ldots, Y_t$ whereas if y_i is a scalar then $\mathbf{y}_t$ is a column vector whose elements are $y_1, \ldots, y_t$. $\uplus_{i=1}^{t} D_i$ defines the multiset summation operation over the sets $D_1, \ldots, D_t$.

Model: At each round t, the agent is given a decision set D_t with K actions, $\hat{x}_{t,1}, \ldots, \hat{x}_{t,K} \in \mathbb{R}^d$. Let V be an $d \times m$ orthonormal matrix with $m \leq d$, where span(V) defines a m-dimensional subspace in $\mathbb{R}^d$. Consider a zero mean true action vector, $x_{t,i} \in \mathbb{R}^d$, such that $x_{t,i} \in$ span(V) for all $i \in [K]$. Let $\psi_{t,i} \in \mathbb{R}^d$ be zero mean random vectors which are

uncorrelated with true action vectors, *i.e.*, $\mathbb{E}[x_{t,i}\psi_{t,i}^T] = 0$ for all $i \in [K]$. Each action vector $\hat{x}_{t,i}$ is generated as follows,

$$\hat{x}_{t,i} = x_{t,i} + \psi_{t,i}. \tag{2.1}$$

This model states that each $\hat{x}_{t,i}$ in D_t is a perturbed version of the true underlying $x_{t,i}$. Denote the covariance matrix of $x_{t,i}$ by Σ_x. Notice that Σ_x is rank-m. Perturbation vectors, $\psi_{t,i}$, are assumed to be isotropic, thus covariance matrix $\Sigma_\psi = \sigma^2 I_d$. Let $\lambda_+ := \lambda_1(\Sigma_x)$ and $\lambda_- := \lambda_m(\Sigma_x)$. The described setting is standard in PCA problems Nadler (2008); Vaswani and Narayanamurthy (2017).

Assumption 1 (Bounded Action and Perturbation Vectors)**.** *There exists finite constants, d_x and d_ψ, such that for all $i \in [K]$,*

$$\|x_{t,i}\|_2^2 \leq d_x \lambda_+, \quad \|\psi_{t,i}\|_2^2 \leq d_\psi \sigma^2.$$

Both d_x and d_ψ can be dependent on m or d and they can be interpreted as the effective dimensions of the corresponding vectors. At each round t, the agent chooses an action, $\hat{X}_t \in D_t$ and observes a reward r_t such that

$$r_t = \hat{X}_t^T \theta_* + \eta_t \qquad \forall t \in [T] \tag{2.2}$$

where $\theta_* \in \text{span}(V)$ is the unknown parameter vector and η_t is the random noise at round t. Notice that since $\theta_* \in \text{span}(V)$, $r_t = \hat{X}_t^T \theta_* + \eta_t = (P\hat{X}_t)^T \theta_* + \eta_t$, where $P = VV^T$ is the projection matrix to the m-dimensional subspace $\text{span}(V)$. We mainly use this expression of the reward in the later parts. Consider $\{F_t\}_{t=0}^\infty$ as any filtration of σ-algebras such that for any $t \geq 1$, $\hat{X}_t$ is F_{t-1} measurable and η_t is F_t measurable.

Assumption 2 (Subgaussian Noise)**.** *For all t, η_t is conditionally R-sub-Gaussian where*

$R \geq 0$ is a fixed constant, ie. $\forall \lambda \in \mathbb{R},\ \mathbb{E}[e^{\lambda \eta_t} | F_{t-1}] \leq e^{\frac{\lambda^2 R^2}{2}}$.

The goal of the agent is to maximize the total expected reward accumulated in any T rounds, $\sum_{t=1}^{T} \hat{X}_t^T \theta_*$. With the knowledge of θ_*, the oracle chooses the action $\hat{X}_t^* = \arg\max_{x \in D_t} x^T \theta_*$ at each round t. We evaluate the agent's performance against the oracle performance. Define *regret* as the difference between expected reward of the oracle and the agent,

$$R_T := \sum_{t=1}^{T} \hat{X}_t^{*T} \theta_* - \sum_{t=1}^{T} \hat{X}_t^T \theta_* = \sum_{t=1}^{T} (X_t^* - \hat{X}_t)^T \theta_*. \tag{2.3}$$

The agent aims to minimize this quantity over time. In the setting described above, the agent is assumed to know that there exists a m-dimensional subspace of $\mathbb{R}^d$ in which true action vectors and the unknown parameter vector lie. This assumption is standard in PCA problems Nadler (2008); Vaswani and Narayanamurthy (2017). In practice these quantities can be estimated and updated in each round. Finally, we define some quantities about the structure of the problem for all $\delta \in (0, 1)$:

$$g_x = \frac{\lambda_+}{\lambda_-},\ g_\psi = \frac{\sigma^2}{\lambda_-}, \Gamma = 2g_\psi + 4\sqrt{g_x g_\psi},\ \alpha = \max(d_x, d_\psi),\ n_\delta = 4\alpha \left(\Gamma \sqrt{\log \frac{2d}{\delta}} + \sqrt{2 g_x \log \frac{m}{\delta}} \right)^2 \tag{2.4}$$

2.3 Overview of PSLB

We propose Projected Stochastic Linear Bandits (PSLB), a SLB algorithm which employs subspace recovery to extract information from the unsupervised data accumulated in the SLB. The PSLB is illustrated in Algorithm 1. PSLB consists of three key elements: subspace estimation, creating confidence sets and acting optimistically. In the following, we will discuss each of them briefly.

Subspace estimation: At each round t, the agent exploits the action vectors observed up

Algorithm 1 PSLB

Input: m, λ_+, λ_-, σ^2, α, δ
for t = 1 to T **do**
  Compute PCA over $\uplus_{i=1}^{t} D_i$
  Create $\hat{P}_t$ with first m eigenvectors
  Construct $\mathcal{C}_{p,t}$, high probability confidence set on $\hat{P}_t$
  Construct $\mathcal{C}_{m,t}$, high probability confidence set for θ_* using subspace recovery
  Construct $\mathcal{C}_{d,t}$, high probability confidence set for θ_* without using subspace recovery
  Construct $\mathcal{C}_t = \mathcal{C}_{m,t} \cap \mathcal{C}_{d,t}$
  $(\tilde{P}_t, \hat{X}_t, \tilde{\theta}_t) = \arg\max_{(P',x,\theta)\in\mathcal{C}_{p,t}\times D_t\times\mathcal{C}_t} (P'x)^T\theta$
  Play $\hat{X}_t$ and observe r_t
end for

to round t, $\uplus_{i=1}^{t} D_i$, to estimate the underlying m-dimensional subspace. In particular, the agent deploys PCA on tK action vectors and computes $\hat{V}_t$, the matrix of top m eigenvectors of $\frac{1}{tK}\sum_{\hat{x}\in\uplus_{i=1}^{t} D_i} \hat{x}\hat{x}^T$. span$(\hat{V}_t)$ is the estimate of the underlying m-dimensional subspace. The agent uses $\hat{V}_t$ to compute $\hat{P}_t := \hat{V}_t\hat{V}_t^T$, the projection matrix onto span$(\hat{V}_t)$, and constructs a high probability confidence set $\mathcal{C}_{p,t}$ around $\hat{P}_t$ which contains both $\hat{P}_t$ and P. In Section 2.4.1 we demonstrate the construction of $\mathcal{C}_{p,t}$, and show that as the agent observes more action vectors, $\mathcal{C}_{p,t}$ shrinks and the estimation error on $\hat{P}_t$ vanishes.

Confidence set construction and optimistic action: At the beginning of each round t, the agent uses $\hat{P}_t$, and projects the supervised actions onto the estimated m-dimensional subspace. The d-dimensional SLB reduces to a m-dimensional SLB problem. The agent then estimates the model parameter θ_*, as θ_t, up to a high probability confidence set $\mathcal{C}_{m,t}$. The tightness of this confidence interval, beside the action-reward pairs, depends on subspace estimation and $\mathcal{C}_{p,t}$.

Simultaneously, relying *only* on the history of action-reward pairs, the agent estimates the model parameter θ_*, as $\hat{\theta}_t$, up to a new high probability confidence set $\mathcal{C}_{d,t}$. This is the same confidence set generation subroutine of OFUL Abbasi-Yadkori et al. (2011). Since θ_* lives in both of these sets with high probability, it lies in the intersection of them with high probability. Finally, the agent takes the intersection of the constructed confidence sets

to create the main confidence set, $\mathcal{C}_t = \mathcal{C}_{m,t} \cap \mathcal{C}_{d,t}$. If an efficient recovery of the subspace is possible, then the plausible parameter set of $\mathcal{C}_{m,t}$ is significantly smaller than the set of $\mathcal{C}_{d,t}$, resulting in smaller $\mathcal{C}_t$ as well as more confident parameter estimation. If the subspace recovery is hard, then $\mathcal{C}_{m,t}$ might not provide much information, and the intersection would mainly result with $\mathcal{C}_{d,t}$.

2.4 Theoretical Analysis of PSLB

In this section, we state the regret upper bound of PSLB and provide the theoretical components that build up to this result. Recalling the quantities defined in (2.4), define Υ such that

$$\Upsilon = \mathcal{O}\left(\left(1 + \Gamma\sqrt{\frac{\alpha}{K}}\right)\left(\frac{\Gamma\sqrt{m\alpha}}{\sqrt{K}\sqrt{\lambda_- + \sigma^2}} + m\right)\right). \tag{2.5}$$

It represents the overall effect of the deploying subspace recovery on the regret in terms of structural properties of SLB setting. It is further discussed in Section 2.4.3. Using Υ, the theorem below states the regret upper bound of PSLB.

Theorem 2.1 (Regret Upper Bound of PSLB)**.** *Fix any $\delta \in (0, 1/6)$. Assume that for all $\hat{x}_{t,i} \in D_t$, $\hat{x}_{t,i}^T\theta_* \in [-1, 1]$. Under Assumptions 1 and 2, $\forall t \geq 1$, with probability at least $1 - 6\delta$, the regret of* PSLB *satisfies*

$$R_t \leq \min\left\{\widetilde{\mathcal{O}}\left(\Upsilon\sqrt{t}\right), \widetilde{\mathcal{O}}\left(d\sqrt{t}\right)\right\}. \tag{2.6}$$

The proof of the theorem involves two main pieces: the projection error analysis (Sections 2.4.1) and the construction of projected confidence sets (Section 2.4.2). Finally, in Section 2.4.3 their role in the proof of Theorem 2.1 is explained and the meaning of the result is

discussed.

2.4.1 Projection Error Analysis

Consider the matrix $\hat{V}_t^T V$ where ith singular value is denoted by $\sigma_i(\hat{V}_t^T V)$, such that $\sigma_1(\hat{V}_t^T V) \geq \ldots \geq \sigma_m(\hat{V}_t^T V)$. Using the analysis in Akhiezer and Glazman (2013), one can show that $\|\hat{P}_t - P\|_2 = \sqrt{1 - \sigma_m^2(\hat{V}_t^T V)} = \sin \Theta_m$, where Θ_m is the largest principal angle between the column spans of V and $\hat{V}_t$. Thus, bounding the projection error between two projection matrices is equivalent to bounding the sine of the largest principal angle between the subspaces that they project. In light of this relation, using Davis-Kahan $\sin \Theta$ theorem (Davis and Kahan, 1970), following lemma bounds the finite sample projection error.

Lemma 1 (Finite Sample Projection Error)**.** *Fix any $\delta \in (0, 1/3)$. Let $t_{w,\delta} = \frac{n_\delta}{K}$. Suppose Assumption 1 holds. Then with probability at least $1 - 3\delta$, $\forall t \geq t_{w,\delta}$,*

$$\|\hat{P}_t - P\|_2 \leq \frac{\phi_\delta}{\sqrt{t}} \quad , \textit{ where } \phi_\delta = 2\Gamma\sqrt{\frac{\alpha}{K}\log\frac{2d}{\delta}}. \tag{2.7}$$

The lemma improves existing bound on the projection error (Corollary 2.9 in Vaswani and Narayanamurthy (2017)) by using the Matrix Chernoff Inequality (Tropp, 2015). It also provides the precise problem dependent quantities in the bound which are required for defining the minimum number of samples required to construct tight confidence sets by using subspace estimation. The general version of the lemma and the details of the proof are given in the Appendix of Lale et al. (2019).

Note that as discussed in Section 2.3, (2.7) defines the confidence set $\mathcal{C}_{p,t}$ for all $t \geq t_{w,\delta}$. Due to equivalence that $\|\hat{P}_t - P\|_2 = \sin \Theta_m$, $\|\hat{P}_t - P\|_2 \leq 1$, $\forall t \geq 1$. Therefore, any projection error bound greater than 1 is vacuous. With the stated $t_{w,\delta}$, the bound on the projection error in (2.7) becomes less than 1 when $t \geq t_{w,\delta}$, with high probability. After round $t_{w,\delta}$,

PSLB starts to produce non-trivial confidence sets $\mathcal{C}_{p,t}$ around $\hat{P}_t$. However, note that $t_{w,\delta}$ can be significantly big for problems that have structure that is hard to recover, e.g. having α linear in d.

Lemma 1 also brings several important intuitions about the subspace estimation problem in terms of the problem structure. Recalling the definition of Γ in (2.4), as g_ψ decreases, the projection error shrinks since the underlying subspace becomes more distinguishable. Conversely, as g_x diverges from 1, it becomes harder to recover the underlying m-dimensional subspace. Additionally, since α is the maximum of the effective dimensions of the true action vector and the perturbation vector, having large α makes the subspace recovery harder and the projection error bound looser, whereas observing more action vectors, K, in each round produces tighter bound on $\|\hat{P}_t - P\|_2$. The effects of these structural properties on the subspace estimation translate to confidence set construction and ultimately to regret upper bound.

2.4.2 Projected Confidence Sets

In this section, we analyze the construction of $\mathcal{C}_{m,t}$ and $\mathcal{C}_{d,t}$. For any round $t \geq 1$, define $\hat{\Sigma}_t := \sum_{i=1}^{t} \hat{X}_i \hat{X}_i^T = \hat{\mathbf{X}}_t \hat{\mathbf{X}}_t^T$. At round t, let $A_t := \hat{P}_t(\hat{\Sigma}_{t-1} + \lambda I_d)\hat{P}_t$ for $\lambda > 0$. The rewards obtained up to round t are denoted as $\mathbf{r}_{t-1}$. At round t, after estimating the projection matrix $\hat{P}_t$ associated with the underlying subspace, PSLB tries to find θ_t, an estimate of θ_*, while believing that θ_* lives within the estimated subspace. Therefore, θ_t is the solution to the following Tikhonov-regularized least squares problem with regularization parameters $\lambda > 0$ and $\hat{P}_t$,

$$\theta_t = \arg\min_{\theta} \|(\hat{P}_t \hat{\mathbf{X}}_{t-1})^T \theta - \mathbf{r}_{t-1}\|_2^2 + \lambda \|\hat{P}_t \theta\|_2^2.$$

Notice that regularization is applied along the estimated subspace. Solving for θ gives $\theta_t = A_t^{\dagger}\big(\hat{P}_t\hat{\mathbf{X}}_{t-1}\mathbf{r}_{t-1}\big)$. Define L such that for all $t \geq 1$ and $i \in [K]$, $\|\hat{x}_{t,i}\|_2 \leq L$ and let $\gamma = \frac{L^2}{\lambda \log\left(1+\frac{L^2}{\lambda}\right)}$. The following theorem gives the construction of projected confidence set, $\mathcal{C}_{m,t}$.

Theorem 2.2 (Projected Confidence Set Construction)**.** *Fix any $\delta \in (0, 1/4)$. Suppose Assumptions 1 & 2 hold, and $\forall t \geq 1$ and $i \in [K]$, $\|\hat{x}_{t,i}\|_2 \leq L$. If $\|\theta_*\|_2 \leq S$ then, with probability at least $1 - 4\delta$, $\forall t \geq t_{w,\delta}$, θ_* lies in the set $\mathcal{C}_{m,t} = \left\{\theta \in \mathbb{R}^d : \|\theta_t - \theta\|_{A_t} \leq \beta_{t,\delta}\right\}$, where*

$$\beta_{t,\delta} = R\sqrt{2\log\left(\frac{1}{\delta}\right) + m\log\left(1+\frac{tL^2}{m\lambda}\right)} + LS\phi_\delta\sqrt{\gamma m\log\left(1+\frac{tL^2}{m\lambda}\right)} + S\sqrt{\lambda}. \tag{2.8}$$

The detailed proof and a general version of the theorem are given in the Appendix of Lale et al. (2019). We will highlight the key aspects in here. The overall proof follows a similar machinery used by Abbasi-Yadkori et al. (2011). Specifically, the first term of $\beta_{t,\delta}$ in (2.8) is derived similarly by using the self-normalized tail inequality. However, since at each round PSLB projects the supervised actions to an estimated m-dimensional subspace to estimate θ_*, d is replaced by m in the bound. While enjoying the benefit of projection, this construction of the confidence set suffers from the finite sample projection error, *i.e.*, uncertainty in the subspace estimation. This effect is observed via second term in (2.8). The second term involves the confidence bound for the estimated projection matrix, ϕ_δ. This is critical in determining the tightness of the confidence set on θ_*. As discussed in Section 2.4.1, ϕ_δ reflects the difficulty of subspace recovery of the given problem and it depends on the underlying structure of the problem and SLB. This shows that as estimating the underlying subspace gets easier, having a projection based approach in the construction of the confidence sets on θ_* provides tighter bounds.

In order to tolerate the possible difficulty of subspace recovery, PSLB also constructs $\mathcal{C}_{d,t}$,

which is the confidence set for θ_* without having subspace recovery. The construction of $\mathcal{C}_{d,t}$ follows OFUL Abbasi-Yadkori et al. (2011). Let $Z_t = \hat{\Sigma}_{t-1} + \lambda I_d$. The algorithm tries to find $\hat{\theta}_t$ which is the ℓ^2-regularized least squares estimate of θ_* in the ambient space. Construction of $\mathcal{C}_{d,t}$ is done under the same assumptions of Theorem 2.2, such that with probability at least $1-\delta$, θ_* lies in the set $\mathcal{C}_{d,t} = \left\{\theta \in \mathbb{R}^d : \|\hat{\theta}_t - \theta\|_{Z_t} \leq \Omega_{t,\delta}\right\}$, where $\Omega_{t,\delta} = R\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1 + \frac{tL^2}{m\lambda}\right)} + S\sqrt{\lambda}$. The search for an optimistic parameter vector happens in $\mathcal{C}_{m,t} \cap \mathcal{C}_{d,t}$. Notice that $\theta_* \in \mathcal{C}_{m,t} \cap \mathcal{C}_{d,t}$ with probability at least $1 - 5\delta$. Optimistically choosing the triplet, $(\tilde{P}_t, \hat{X}_t, \tilde{\theta}_t)$, within the described confidence sets gives PSLB a way to tolerate the possibility of failure in recovering an underlying structure. If confidence set $\mathcal{C}_{m,t}$ is loose or PSLB is not able to recover an underlying structure, then $\mathcal{C}_{d,t}$ provides the useful confidence set to obtain desirable learning behavior.

2.4.3 Regret Analysis

Using the intersection of $\mathcal{C}_{m,t}$ and $\mathcal{C}_{d,t}$ as the confidence set at round t, gives PSLB the ability to obtain the lowest possible instantaneous regret among both confidence sets. Therefore, the regret of PSLB is upper bounded by the minimum of the regret upper bounds on the individual strategies. Using only $\mathcal{C}_{d,t}$ is equivalent to following OFUL and the regret analysis can be found in Abbasi-Yadkori et al. (2011). The regret analysis of using only the projected confidence set $\mathcal{C}_{m,t}$ is the main contribution of this work.

The derivation of the regret upper bound can be found in the Appendix of Lale et al. (2019). Here we elaborate more on the nature of the regret obtained by using $\mathcal{C}_{m,t}$ only, *i.e.* first term in Theorem 2.1, and discuss the effect and meaning of Υ in particular.

Υ is the reflection of the finite sample projection error at the beginning of the algorithm. It captures the difficulty of subspace recovery based on the structural properties of the problem and determines the regret of deploying projection based methods in SLBs. Recall that α

is the maximum of the effective dimensions of the true action vectors and the perturbation vectors. Depending on the structure of the problem, α can be $\mathcal{O}(d)$, e.g., the perturbation can be uniform all dimensions, which prevents the projection error from shrinking; thus, causes $\Upsilon = \mathcal{O}(d\sqrt{m})$ resulting in $\widetilde{\mathcal{O}}(d\sqrt{mt})$ regret. The eigengap within the true action vectors g_x and the eigengap between the true action vectors and the perturbation vectors g_ψ are critical factors that determine the identifiability of the hidden subspace. As σ^2 increases, the subspace recovery becomes harder since the effect of perturbation increases. Conversely, as λ_- increases, the underlying subspace becomes easier to identify. These effects are significant and translate to regret of PSLB via Γ in Υ.

Moreover, having finite samples to estimate the subspace affects the regret bound through Υ. Due to the nature of SLB, *i.e.* finite action vectors in decision sets, this is unavoidable. Note that if we were given infinitely many actions in the decision set, the subspace recovery would be accomplished perfectly. Thus, in the setting of $K \to \infty$, the problem becomes m-dimensional SLB having regret upper bound of $\widetilde{\mathcal{O}}(m\sqrt{t})$, since $\Upsilon = \mathcal{O}(m)$ as $K \to \infty$. Overall, with all these components, Υ represents the hardness of using PCA based methods in dimensionality reduction in SLBs.

Theorem 2.1 states that if the underlying structure is easily recoverable, e.g. $\Upsilon = \mathcal{O}(m)$, then using PCA based dimension reduction and construction of confidence sets provide substantially better regret upper bound for large d. If that is not the case, then due to the best of the both worlds approach provided by PSLB, the agent obtains the best possible regret upper bound. Note that the bound for using only $\mathcal{C}_{m,t}$ is a worst case bound and as we present in Section 2.5, in practice PSLB can give significantly better results.

(a) **(b)**

Figure 2.1: **(a)** 2-D representation of the effect of increasing perturbation level in concealing the underlying subspace **(b)** Regrets of PSLB vs. OFUL under $d_\psi = 1, 10$ and 20. As the effect of perturbation increases PSLB's performance approaches to performance of OFUL

2.5 Experiments

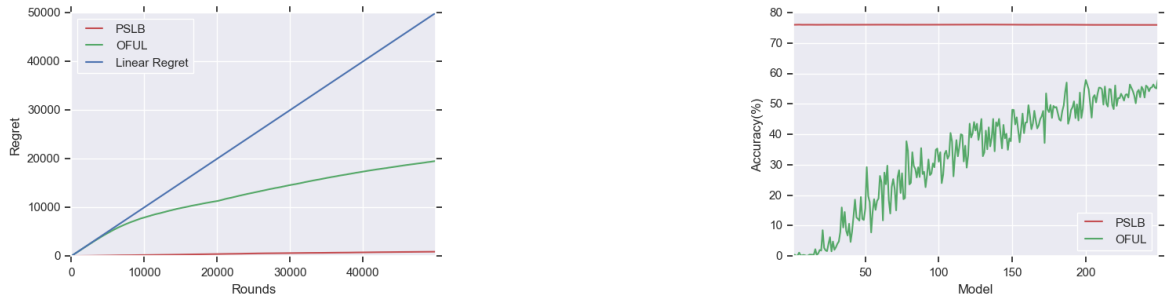
Synthetic example: We study PSLB on 50 dimensional SLBs with 4 dimensional hidden subspace structure. At each round t, there are $K = 200$ actions in D_t. Each action is generated as $\hat{x}_{t,i} = x_{t,i} + \psi_{t,i}$. $\psi_{t,i} \in \mathcal{R}^d$ is drawn from Normal distribution but rejected if $\|\psi_{t,i}\|_2^2 > d_\psi$. We picked an orthonormal matrix $V \in \mathcal{R}^{50\times 4}$ and generate $x_{t,i}$ such that $x_{t,i} = V\epsilon$ where $\epsilon \sim uniform([-1, 1])^4$. For $T = 10,000$ rounds, we generate 3 different decision sets using $d_\psi = 1, 10$ and 20.

As depicted in 2-D representation in Figure 2.1(a), increasing the perturbation level conceals the hidden structure resulting in harder subspace recovery. Using these SLB settings, we studied the performance of PSLB and OFUL. Figure 2.1(b) provides the change in regrets as we increase the noise level from $d_\psi = 1$ to $d_\psi = 20$. Note that d_ψ can be interpreted as the effective dimension of the perturbation vectors. As the d_ψ increases the perturbations become more dominant, PSLB loses its advantage of recovering the underlying subspace and starts performing similar to OFUL. As suggested in the analysis, having d_ψ close to dimension of the ambient space leads to poor subspace recovery performance, higher regret in SLBs. This example demonstrates the overall effect of perturbation level on the subspace estimation, confidence set construction and ultimately regret.

Image Classification in SLB Setting: In the image classification experiments, we study MNIST, CIFAR-10 and ImageNet datasets and use them to create the decision sets for the SLB setting. We train standard DNNs on each dataset to generate the feature representations of each image for each class and use these features as the decision sets at each time step of SLB. In other words, for all images in the datasets DNNs generate an action (label) representation for every class. Thus, we obtain 10 action vectors for each image in MNIST and CIFAR-10, and 1000 action vectors for ImageNet. In the SLB setting, the agent receives a reward of 1 if it chooses the right action, which is the label representation for the correct class according to trained network, and 0 otherwise. We apply both PSLB and OFUL on these SLBs. We measure the regret by counting the number of mistakes each algorithm makes. For details of experimental setting please refer the Appendix of Lale et al. (2019).

Through computing PCA of the empirical covariance matrix of the action vectors, surprisingly we found that projecting action vectors onto the 1-dimensional subspace defined by the dominant eigenvector is sufficient for these datasets in the SLB setting; thus, $m = 1$. While surprising, a similar observation is founded by Chaudhari and Soatto (2018) that the diffusion matrix which depends on the architecture, weights and the dataset has significantly low rank structure for MNIST and CIFAR-10 datasets. We present the regret obtained by PSLB and OFUL for ImageNet with $d = 100$ in Figure 2.2(a). During the experiment, PSLB tries to recover a 1-dimensional subspace using the action vectors collected.

With the help of subspace recovery and projection, PSLB provides a massive reduction in the dimensionality of the SLB problem and immediately estimates a fairly accurate model for θ_*. On the other hand, OFUL naively tries to sample from all dimensions in order to learn θ_*. This difference yields orders of magnitude improvement in regret. During the SLB experiment, we also sample the optimistic models that are chosen by PSLB and OFUL. We use these models to test the model accuracy of the algorithms, *i.e.* perform classification over all images in dataset. The optimistic model accuracy comparison is depicted in Figure 2.2(b).



(a) **(b)**

Figure 2.2: **(a)** Regret of PSLB vs. OFUL in SLB setting with ImageNet for $d = 100$ **(b)** Image classification accuracy of periodically sampled optimistic models of PSLB and OFUL on ImageNet

This portrays the learning behavior of PSLB and OFUL. Using projection, PSLB learns the underlying linear model in the first few rounds, whereas OFUL suffers from high-dimension of SLB framework and lack of knowledge besides chosen action-reward pairs. The Appendix in Lale et al. (2019) provides an extensive study of all datasets with different settings.

2.6 Related Work

The study of linear bandit problems extends to various algorithms and environment settings (Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010; Li et al., 2010). Kleinberg et al. (2010) studies the class of problems when the decision set changes time to time, while Dani et al. (2008) studies this problem when the decision set provides a set of fixed actions. Further analysis in the area extend these approaches to classes where there are more structures in the problem setup, e.g., graph structure inspired by social media (Valko et al., 2014). In traditional decision-making problems, where hand engineered feature representations are provided, sparsity in the linear function is a valid structure. Sparsity, as the key in high-dimensional conventional structured linear bandits, conveys series of successes in classical settings (Abbasi-Yadkori et al., 2012; Carpentier and Munos, 2012). In recommendations systems, where a set of users and items are given, Gopalan et al. (2016) considers the low-

rank structure of the user-item preference matrix and provide an algorithm which exploits this further structure.

Subspace recovery and dimensionality reduction problems are well studied in the literature. Several linear and nonlinear dimension reduction methods have been proposed such as PCA (Pearson, 1901), independent component analysis (Hyvärinen and Oja, 2000), random projections (Candes and Tao, 2006) and non-convex robust PCA (Netrapalli et al., 2014). Among the linear dimension reduction techniques, PCA is the simplest, yet most widely used method. Analysis of PCA based methods mostly focus on the asymptotic results (Anderson et al., 1963; Jain et al., 2016). However, in the settings like SLB with finite number of actions, it is necessary to have finite sample guarantees. In the literature, among few finite sample PCA works, Nadler (2008) provides finite sample guarantees for one-dimensional PCA, whereas Vaswani and Narayanamurthy (2017) extends it to larger dimensions with various noise models.

2.7 Conclusion

In this paper, we study high dimensional SLB problems with hidden low rank structure. We propose PSLB, an efficient SLB algorithm which utilizes subspace recovery methods to estimate the effective subspace and enhance the regret upper bound on SLB problems. While PSLB is not limited to any particular subspace recovery method, we choose PCA for this study. Furthermore, for the exploration/exploitation, we deploy optimism while any other efficient exploration strategy is also applicable. We theoretically show that if such linear structure does not exist or is hard to recover, then the PSLB reduces to the standard SLB algorithm, OFUL. We empirically study MNIST, CIFAR-10, and ImageNet datasets to have image classification task in SLB framework. We test the performance of PSLB vs. OFUL. We show that when DNNs produce features of the actions, a significantly low

dimensional structure is observed. Due to this structure, we showed that PSLB substantially outperforms OFUL and converges to an accurate model while OFUL still struggles to sample in high dimensions to learn the underlying parameter vector.

In the future work, we plan to extend this line of study to the general class of low dimensional manifold structured problems. Bora et al. (2017) peruses a similar approach for compression problems. While optimism is the primary approach in the theoretical analyses of SLBs, it poses a computationally intractable internal optimization problem. An alternative method is Thompson sampling, a practical algorithm for SLBs. In future work, we plan to deploy Thompson sampling and mitigate the computational complexity of PSLB.

Chapter 3

RL in Markov Decision Processes

Efficient Exploration through Bayesian Deep Q-Networks

We study reinforcement learning (RL) in high dimensional episodic Markov decision processes (MDP). We consider value-based RL when the optimal Q-value is a linear function of d-dimensional state-action feature representation. For instance, in deep-Q networks (DQN), the Q-value is a linear function of the feature representation layer (output layer). We propose two algorithms, one based on optimism, LinUCB, and another based on posterior sampling, LinPSRL. We guarantee frequentist and Bayesian regret upper bounds of $\widetilde{\mathcal{O}}(d\sqrt{T})$ for these two algorithms, where T is the number of episodes. We extend these methods to deep RL and propose Bayesian deep Q-networks (BDQN), which uses an efficient Thompson sampling algorithm for high dimensional RL. We deploy the double DQN (DDQN) approach, and instead of learning the last layer of Q-network using linear regression, we use Bayesian linear regression, resulting in an approximated posterior over Q-function. This allows us to directly incorporate the uncertainty over the Q-function and deploy Thompson sampling on the learned posterior distribution resulting in efficient exploration/exploitation trade-off. We empirically study the behavior of BDQN on a wide range of Atari games. Since BDQN

carries out more efficient exploration and exploitation, it is able to reach higher return substantially faster compared to DDQN.

3.1 Introduction

One of the central challenges in reinforcement learning (RL) is to design algorithms with efficient exploration-exploitation trade-off that scale to high-dimensional state and action spaces. Recently, deep RL has shown significant promise in tackling high-dimensional (and continuous) environments. These successes are mainly demonstrated in simulated domains where exploration is inexpensive and simple explorations-exploitation approaches such as ε-greedy or Boltzmann strategies are deployed. ε-greedy chooses the most greedy action with $1 - \varepsilon$ probability and randomizes over all the actions, and does not consider the estimated Q-values or its uncertainties. The Boltzmann strategy considers the estimated Q-values to guide the decision making but still does not exploit its uncertainty in estimation. For complex environments, more statistically efficient strategies are required. One such strategy is optimism in the face of uncertainty (OFU), where we follow the decision suggested by the most optimistic estimation of the environment and guarantee efficient exploration/exploitation strategies. Despite compelling theoretical results, these methods are mainly model based and limited to tabular settings (Jaksch et al., 2010a; Auer, 2003).

An alternative to OFU is posterior sampling (PS), or more general randomized approach, is Thompson Sampling (Thompson, 1933) which, under the Bayesian framework, maintains a posterior distribution over the environment model, see Table 3.1. Thompson sampling has shown strong performance in many low dimensional settings such as multi-arm bandits (Chapelle and Li, 2011) and small tabular MDPs (Osband et al., 2013). Thompson sampling requires sequentially sampling of the models from the (approximate) posterior or uncertainty and to act according to the sampled models to trade-off exploration and ex-

ploitation. However, the computational costs in posterior computation and planning become intractable as the problem dimension grows.

To mitigate the computational bottleneck, Osband et al. (2014) consider episodic and tabular MDPs where the optimal Q-function is linear in the state-action representation. They deploy Bayesian linear regression (BLR) (Rasmussen and Williams, 2006) to construct an approximated posterior distributing over the Q-function and employ Thompson sampling for exploration/exploitation. The authors guarantee an order optimal regret upper bound on the tabular MDPs in the presence of a Dirichlet prior on the model parameters. Our paper is a high dimensional and general extension of (Osband et al., 2014).

Table 3.1: Thompson Sampling, similar to OFU and PS, incorporates the estimated Q-values, including the greedy actions, and uncertainties to guide exploration-exploitation trade-off. ε-greedy and Boltzmann exploration fall short in properly incorporating them. ε-greedy consider the most greedy action, and Boltzmann exploration just exploit the estimated returns. Full discussion in Appendix of (Azizzadenesheli and Anandkumar, 2018).

Strategy	Greedy-Action	Estimated Q-values	Estimated uncertainties
ε-greedy	✓	✗	✗
Boltzmann exploration	✓	✓	✗
Thompson Sampling	✓	✓	✓

While the study of RL in general MDPs is challenging, recent advances in the understanding of linear bandits, as an episodic MDPs with episode length of one, allows tackling high dimensional environment. This class of RL problems is known as LinReL. In linear bandits, both OFU (Abbasi-Yadkori et al., 2011; Li et al., 2010) and Thompson sampling (Russo and Van Roy, 2014b; Agrawal and Goyal, 2013; Abeille and Lazaric, 2017) guarantee promising results for high dimensional problems. In this paper, we extend LinReL to MDPs.

Contribution 1 – Bayesian and frequentist regret analysis: We study RL in episodic MDPs where the optimal Q-function is a linear function of a d-dimensional feature representation of state-action pairs. We propose two algorithms, LinPSRL, a Bayesian method using PS, and LinUCB, a frequentist method using OFU. LinPSRL constructs a posterior

distribution over the linear parameters of the Q-function. At the beginning of each episode, LinPSRL draws a sample from the posterior then acts optimally according to that model. LinUCB constructs the upper confidence bound on the linear parameters and in each episode acts optimally with respect to the most optimistic model. We provide theoretical performance guaranteess and show that after T episodes, the Bayesian regret of LinPSRL and the frequentist regret of LinUCB are both upper bounded by $\widetilde{\mathcal{O}}(d\sqrt{T})$[1].

Contribution 2 – From theory to practice: While both LinUCB and LinPSRL are statistically designed for high dimensional RL, their computational complexity can make them practically infeasible, e.g., maintaining the posterior can become intractable. To mitigate this shortcoming, we propose a unified method based on the BLR approximation of these two methods. This unification is inspired by the analyses in Abeille and Lazaric (2017); Abbasi-Yadkori et al. (2011) for linear bandits. 1) For LinPSRL: we deploy BLR to approximate the posterior distribution over the Q-function using conjugate Gaussian prior and likelihood. In tabular MDP, this approach turns out to be similar to Osband et al. (2014)[2]. 2) For LinUCB: we deploy BLR to fit a Gaussian distribution to the frequentist upper confidence bound constructed in OFU (Fig 2 in Abeille and Lazaric (2017)). These two approximation procedures result in the same Gaussian distribution, and therefore, the same algorithm. Finally, we deploy Thompson sampling on this approximated distribution over the Q-functions. While it is clear that this approach is an approximation to PS, Abeille and Lazaric (2017) show that this approach is also an approximation to OFU [3]. For practical use, we extend this unified algorithm to deep RL, as described below.

Contribution 3 – Design of BDQN: We introduce Bayesian Deep Q-Network (BDQN), a Thompson sampling based deep RL algorithm, as an extension of our theoretical development to deep neural networks. We follow the DDQN (Van Hasselt et al., 2016) architecture and

[1] The dependency in the episode length is more involved and details are in Section 3.2.
[2] We refer the readers to this work for an empirical study of BLR on tabular environment.
[3] An extra $\sqrt{d}$ expansion of the Gaussian approximation is required for the theoretical analysis.

train the Q-network in the same way except for the last layer (the linear model) where we use BLR instead of linear regression. We deploy Thompson sampling on the approximated posterior of the Q-function to balance between exploration and exploitation. Thus, BDQN requires a simple and a minimal modification to the standard DDQN implementation.

We empirically study the behavior of BDQN on a wide range of Atari games (Bellemare et al., 2013; Machado et al., 2017). Since BDQN follows an efficient exploration-exploitation strategy, it reaches much higher cumulative rewards in fewer interactions, compared to its ε-greedy predecessor DDQN. We empirically observed that BDQN achieves DDQN performance in less than 5M$\pm$1M interactions for almost half of the games while the cumulative reward improves by a median of 300% with a maximum of $80K\%$ on all games. Also, BDQN has $300\% \pm 40\%$ (mean and standard deviation) improvement over these games on the area under the performance measure. Thus, BDQN achieves better sample complexity due to a better exploration/exploitation trade-off.

Comparison: Recently, many works have studied efficient exploration/exploitation in high dimensional environments. (Lipton et al., 2016b) proposes a variational inference-based approach to help the exploration. Bellemare et al. (2016) proposes a surrogate for optimism. Osband et al. (2016) proposes an ensemble of many DQN models. These approaches are significantly more expensive than DDQN (Osband et al., 2016) while also require a massive hyperparameter tuning effort (Bellemare et al., 2016).

In contrast, our approach has the following desirable properties: 1) **Computation**: BDQN nearly has a same computational complexity as DDQN since there is no backpropagation in the last layer of BDQN (therefore faster), but instead, there is a BLR update which requires inverting a small 512×512 matrix (order of less than a second), once in a while. 2) **Hyperparameters**: no exhaustive hyper-parameter tuning. We spent less than two days of academic level GPU time on hyperparameter tuning of BLR in BDQN which is another evidence on its significance. 3) **Reproducibility**: All the codes, with detailed comments

and explanations, are publicly available.

3.2 Linear Q-function

3.2.1 Preliminaries

Consider an episodic MDP $M := \langle \mathcal{X}, \mathcal{A}, P, P_0, R, \gamma, H \rangle$, with horizon length H, state space $\mathcal{X}$, closed action set $\mathcal{A}$, transition kernel P, initial state distribution P_0, reward distribution R, discount factor $0 \leq \gamma \leq 1$. For any natural number H, $[H] = \{1, 2, \ldots, H\}$. The time step withing the episode, h, is encoded in the state, i.e., $\mathcal{X}^h$ and $\mathcal{A}^h$, $\forall h \in [H]$. We drop h in state-action definition for brevity. $\|\cdot\|_2$ denotes the spectral norm and for any positive definite matrix χ, $\|\cdot\|_\chi$ denotes the χ matrix-weighted spectral norm. At a given time step h, we define agent's Q-function at state x^h, as an agent's expected return after taking action a^h and then following a policy π.

$$Q(x^h, a^h) = \mathbb{E}\left[R(x^h, a^h) + \gamma Q(x^{h+1}, \pi(x^{h+1}))\middle| x^h, a^h\right]$$

Following the Bellman optimality in MDPs, we have that for the optimal Q-function

$$Q^*(x^h, a^h) = \mathbb{E}\left[R(x^h, a^h) + \gamma \max_{a \in \mathcal{A}} Q^*(x^{h+1}, a)\middle| x^h, a^h\right]$$

We consider MDPs where the optimal Q-function, similar to linear bandits, is linear in state-action representations $\phi(\cdot, \cdot) := \mathcal{X} \times \mathcal{A} \to \mathcal{R}^d$, i.e., $Q^{\omega^*}_{\pi^*}(x^h, a^h) := \phi(x^h, a^h)^\top \omega^{*h}$, $\forall x^h, a^h \in \mathcal{X} \times \mathcal{A}$. ω^* denotes the set of $\omega^{*h} \in \mathbb{R}^d$ $\forall h \in [H]$, representing the environment and π^* the set of $\pi^{*h} : \mathcal{X} \to \mathcal{A}$ with $\pi^{*h}(x) := \arg\max_{a \in \mathcal{A}} Q^{\omega^{*h}}_{\pi^*}(x^h, a^h)$. $V^{\omega^*}_{\pi^*}$ denotes the corresponding value function.

Algorithm 2 LinPSRL

```
Input: the prior and likelihood
for episode: t= 1,2,... do
    ω_t^h ~ posterior distribution, ∀h ∈ [H]
    for h = 0 to the end of episode do
        Follow π_t induced by ω_t^h
    end for
    Update the posterior
end for
```

Algorithm 3 LinUCB

```
Input: σ, λ and δ
for episode: t = 1,2,... do
    for h = 1 to the end of episode do
        choose optimistic ω̃_t^h in C_{t-1}^h(δ)
        Follow π̃_t^h induced by ω̃_t^h
    end for
    Compute the confidence C_t^h(δ), ∀h ∈ [H]
end for
```

3.2.2 LinReL

At each time step h and state action pair x^h, a^h, define ν^h a mean zero random variable that captures stochastic reward and transition at time step h;

$$\phi(x^h, a^h)^\top \omega^{*h} + \nu^h = R^h + \gamma\phi(x^{h+1}, \pi^{*h+1}(x^{h+1}))^\top \omega^{*h+1}$$

where R^h is the reward at time step h. Definition of ν^h plays an important role since knowing ω^{*h+1} and π^{*h+1} reduces the learning of ω^{*h} to the standard Martingale based linear regression problem. Of course we neither have ω^{*h+1} nor have π^{*h+1}.

LinPSRL(Algorithm 2): In this Bayesian approach, the agent maintains the prior over the vectors $\omega^{*h}, \forall h$ and given the collected experiences, updates their posterior at the beginning of each episode. At the beginning of each episode t, the agent draws $\omega_t^h, \forall h$, from the posterior, and follows their induced policy π_t^h, i.e., $a_t^h := \arg\max_{a\in\mathcal{A}} \phi^\top(x^h, a)\omega_t^h, \forall x^h \in \mathcal{X}$.

LinUCB(Algorithm 3): In this frequentist approach, at the beginning of t'th episode, the agent exploits the so-far collected experiences and estimates ω^{*h} up to a high probability confidence intervals $\mathcal{C}_{t-1}^h$ i.e., $\omega^{*h} \in C_{t-1}^h$, $\forall h$. At each time step h, given a state x_t^h, the agent follows the optimistic policy; $\widetilde{\pi}_t^h(x_t^h) = \arg\max_{a\in\mathcal{A}} \max_{\omega\in\mathcal{C}_{t-1}^h} \phi^\top(X_t^h, a)\omega$.

Regret analysis: For both of these approaches, we show that, as we get more samples,

the confidence sets $\mathcal{C}_t^h$, $\forall h$, shrink with the rate of $\widetilde{\mathcal{O}}\left(1/\sqrt{t}\right)$, resulting in more confidence parameter estimation and therefore smaller per step regret (Appendix of (Azizzadenesheli and Anandkumar, 2018)). For linear models, define the gram matrix χ_t^h and also ridge regularized matrix with $\widetilde{\chi}^h \in \mathbb{R}^{d\times d}$ (we set it to λI)

$$\chi_t^h := \sum_{i=1}^{t} \phi_i^h {\phi_i^h}^\top, \qquad \overline{\chi}_t^h = \chi_t^h + \widetilde{\chi}^h$$

Following the standard assumption in the self normalized analysis of linear regression and linear bandit (Peña et al., 2009; Abbasi-Yadkori et al., 2011), we have;

- $\forall h \in [H]$: the noise vector ν^h is a σ-sub-Gaussian vector. (refer to Assumption in Appendix of (Azizzadenesheli and Anandkumar, 2018))
- $\forall h \in [H]$ we have $\|\omega^{*h}\|_2 \leq L_\omega$, $\|\phi(x^h, a^h)\phi(x^h, a^h)^\top\|_2^2 \leq L$, $\forall x \in \mathcal{X}, a \in \mathcal{A}$, a.s.
- Expected rewards and returns are in $[0, 1]$.

Then, $\forall h$ define ρ_λ^h such that;

$$\sum_{i}^{t} \|\phi(x_i^h, \pi^*(x_i^h))\|^2_{\overline{\chi}_t^{h-1}} \leq \rho_\lambda^h, \ \forall h, t, \textit{with } \rho_\lambda^{H+1} = 0$$

similar to the ridge linear regression analysis in Hsu et al. (2012), we require $\rho_\lambda^h < \infty$. This requirement is automatically satisfied if the optimal Q-function is bounded away from zero (all features have large component at least in one direction). Let $\overline{\rho}_\lambda^H(\gamma)$ denote the following combination of ρ_λ^h;

$$\overline{\rho}_\lambda^H(\gamma) := \sum_{i=1}^{H} (\gamma)^{H-i} \left(\frac{1}{H} + \frac{1}{H} \sum_{j=1}^{i} \prod_{k=1}^{j} (\gamma)^j \rho_\lambda^{H-(i-k)+1} \right)$$

For any prior and likelihood satisfying these assumptions, we have;

Theorem 3.1 (Bayesian Regret)**.** *For an episodic MDP with episode length H, discount factor γ, and feature map $\phi(x,a) \in \mathbb{R}^d$, after T episodes the Bayesian regret of* LinPSRL *is upper bounded as;*

$$\boldsymbol{BayesReg}_T = \mathbb{E}\left[\sum_t^T \left[V_{\pi^*}^{\omega^*} - V_{\widetilde{\pi}_t}^{\omega^*}\right]\right] = \mathcal{O}\left(d\sqrt{\overline{\rho}_\lambda^H(\gamma)HT}\log(T)\right)$$

Proof is given in the Appendix of (Azizzadenesheli and Anandkumar, 2018).

Theorem 3.2 (Frequentist Regret)**.** *For an episodic MDP with episode length H, discount factor γ, feature map $\phi(x,a) \in \mathbb{R}^d$, after T episodes the frequentist regret of* LinUCB *is upper bounded as;*

$$\boldsymbol{Reg}_T := \mathbb{E}\left[\sum_t^T \left[V_{\pi^*}^{\omega^*} - V_{\widetilde{\pi}_t}^{\omega^*}\right] \middle| \omega^*\right] = \mathcal{O}\left(d\sqrt{\overline{\rho}_\lambda^H(\gamma)HT}\log(T)\right)$$

Proof is given in the Appendix of (Azizzadenesheli and Anandkumar, 2018). These regret upper bounds are similar to those in linear bandits (Abbasi-Yadkori et al., 2011; Russo and Van Roy, 2014b) and linear quadratic control (Abbasi-Yadkori and Szepesvári, 2011), i.e. $\widetilde{\mathcal{O}}(d\sqrt{T})$. Since linear bandits are special cases of episodic continuous MDPs, when horizon is equal to $H = 1$, we observe that our Bayesian regret upper bound recovers (Russo and Van Roy, 2014b) and our frequentist regret upper bound recovers the bound in (Abbasi-Yadkori et al., 2011). While our regret upper bounds are order optimal in T, and d, they have bad dependency in the horizon length H. In our future work, we plan to extensively study this problem and provide tight lower and upper bound in terms of T, d and H.

3.3 Bayesian Deep Q-Networks

We propose Bayesian deep Q-networks (BDQN) an efficient Thompson sampling based method in high dimensional RL problems. In value based RL, the core of most prominent approaches is to learn the Q-function through minimizing a surrogate to Bellman residual (Schweitzer and Seidmann, 1985; Lagoudakis and Parr, 2003; Antos et al., 2008) using temporal difference (TD) update (Tesauro, 1995). Van Hasselt et al. (2016) carries this idea, and propose DDQN (similar to its predecessor DQN (Mnih et al., 2015)) where the Q-function is parameterized by a deep network. DDQN employ a target network Q^{target}, target value $y = r + \gamma Q^{target}(x', \hat{a})$, where the tuple (x, a, r, x') are consecutive experiences, $\hat{a} = \arg\max_{a'} Q(x', a')$. DDQN learns the Q function by approaching the empirical estimates of the following regression problem:

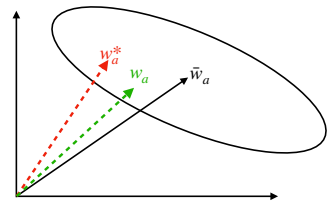


Figure 3.1: BDQN deploys Thompson Sampling to $\forall a \in A$ sample w_a (therefore a Q-function) around the empirical mean $\overline{w}_a$ and w_a^* the underlying parameter of interest.

$$\mathcal{L}(Q, Q^{target}) = \mathbb{E}\left[(Q(x, a) - y)^2\right] \tag{3.1}$$

The DDQN agent, once in a while, updates the Q^{target} network by setting it to the Q network, and follows the regression in Eq.3.1 with the new target value. Since we aim to empirically study the effect of Thompson sampling, we directly mimic the DDQN to design BDQN.

Linear Representation: DDQN architecture consists of a deep neural network where the Q-value is a linear function of the feature representation layer (output layer) of the Q-network, i.e., $\phi_\theta(x) \in \mathbb{R}^d$ parameterized by θ. Therefore, for any x, a, $Q(x, a) = \phi_\theta(x)^\top w_a$ with $w_a \in \mathcal{R}^d$, the parameter of the last linear layer. Similarly, the target model has the same architecture, and consists of $\phi_{\theta^{target}}(\cdot)$, the feature representation of the target network, and ${w^{target}}_a$, $\forall a \in \mathcal{A}$ the target weight. Given a tuple (x, a, r, x') and $\hat{a} = \arg\max_{a'} {\phi_\theta}^\top w_{a'}$,

DDQN learns w_a's and θ to match y:

$$Q(x,a) = \phi_\theta(x)^\top w_a \rightarrow y := r + \gamma \phi_{\theta^{target}}(x')^\top w^{target}{}_{\hat{a}}$$

In DDQN, we match $\phi_\theta(x)^\top w_a$ to y using the regression in Eq. 3.1. This regression problem results in a linear regression in the last layer, w_a's. BDQN follows all DDQN steps except for the learning of the last layer w_a's. BDQN deploys Gaussian BLR instead of the plain linear regression, resulting in an approximated posterior on the w_a's and consequently on the Q-function. As discussed before, BLR with Gaussian prior and likelihood is an approximation to LinPSRL and LinUCB (Abeille and Lazaric, 2017). Through BLR, we efficiently approximate the distribution over the Q-values, capture the uncertainty over the Q estimates, and design a efficient exploration-exploitation strategy using Thompson Sampling.

Given a experience replay buffer $\mathcal{D} = \{x_\tau, a_\tau, y_\tau\}_{\tau=1}^{D}$, for each action a we construct a data set $\mathcal{D}_a$ with $a_\tau = a$, then construct a matrix $\Phi_a^\theta \in \mathbb{R}^{d \times |\mathcal{D}_a|}$, the concatenation of feature vectors $\{\phi_\theta(x_i)\}_{i=1}^{|\mathcal{D}_a|}$, and $\mathbf{y}_a \in \mathbb{R}^{|\mathcal{D}_a|}$, the concatenation of target values in set $\mathcal{D}_a$. We then approximate the posterior distribution of w_a as follows:

$$\overline{w}_a := \frac{1}{\sigma_\epsilon^2} Cov_a \Phi_a^\theta \mathbf{y}_a, \quad Cov_a := \left(\frac{1}{\sigma_\epsilon^2} \Phi_a^\theta {\Phi_a^\theta}^\top + \frac{1}{\sigma^2} I \right)^{-1} \rightarrow \textit{sampling } w_a \sim \mathcal{N}\left(\overline{w}_a, Cov_a\right) \tag{3.2}$$

which is the derivation of well-known BLR. Fig. 3.1 demonstrate the mean and covariance of the over w_a for each action a. A BDQN agent deploys Thompson sampling on the approximated posteriors every T^S to balance exploration and exploitation while updating the posterior every T^{BT}.

Algorithm 4 BDQN

Initialize θ, θ^{target}, and $\forall a$, w_a, w_a^{target}, Cov_a
Set the replay buffer $RB = \{\}$
for $t = 1,2,3\ldots$ **do**
  if $t \bmod T^{BT} = 0$ **then**
    $\forall a$, update w_a^{target} and Cov_a, $\forall a$
  end if
  if $t \bmod T^{S} = 0$ **then**
    Draw $w_a \sim \mathcal{N}\left(w_a^{target}, Cov_a\right)\ \forall a$
  end if
  Set $\theta^{target} \leftarrow \theta$ every T^T
  Execute $a_t = \arg\max_{a'} w_{a'}^{\top} \phi_\theta(x_t)$
  Store (x_t, a_t, r_t, x_{t+1}) in the RB
  Sample a minibatch $(x_\tau\ ,\ a_\tau,\ r_\tau,\ x_{\tau+1})$ from the RB
  $y_\tau \leftarrow \left\{ \begin{array}{ll} r_\tau & \text{terminal } x_{\tau+1} \\ r_\tau + w_{\hat{a}}^{target\top} \phi_{\theta^{target}}(x_{\tau+1}),\ \hat{a} := \arg\max_{a'} w_{a'}^{\top} \phi_\theta(x_{\tau+1}) & \text{non-terminal } x_{\tau+1} \end{array} \right\}$
  Update $\theta \leftarrow \theta - \alpha \cdot \nabla_\theta (y_\tau - w_{a_\tau}^{\top} \phi_\theta(x_\tau))^2$
end for

3.4 Experiments

We empirically study BDQN behaviour on a variety of Atari games in the Arcade Learning Environment (Bellemare et al., 2013) using OpenAI Gym (Brockman et al., 2016). All the codes, with detailed comments and explanations are publicly available and programmed in MxNet (Chen et al., 2015). We evaluate BDQN on the measures of sample complexity and score against DDQN, Fig 3.2.

We implemented DDQN and BDQN following Van Hasselt et al. (2016). We also attempted to implement a few other deep RL methods that employ strategic exploration (with advice from their authors), e.g., (Osband et al., 2016; Bellemare et al., 2016). Unfortunately we encountered several implementation challenges that we could not address since neither the codes nor the implementation details are publicly available (we were not able to reproduce their results beyond the performance of random policy). Along with BDQN and DDQN codes, we also made our implementation of Osband et al. (2016) publicly available. In order to illustrate the BDQN performance we report its scores along with a number of state-of-

the-art deep RL methods 3.2. For some games, e.g., Pong, we ran the experiment for a longer period but just plotted the beginning of it in order to observe the difference. Due to huge cost of deep RL methods, for some games, we run the experiment until a plateau is reached. The BDQN and DDQN columns are scores after running them for number steps reported in *Step* column without Since the regret is considered, no evaluation phase designed for them. DDQN$^+$ is the reported scores of DDQN in Van Hasselt et al. (2016) at evaluation time where the $\varepsilon = 0.001$. We also report scores of Bootstrap DQN (Osband et al., 2016), NoisyNet (Fortunato et al., 2017), CTS, Pixel, Reactor (Ostrovski et al., 2017). For NoisyNet, the scores of NoisyDQN are reported. To illustrate the sample complexity behavior of BDQN we report SC: the number of interactions BDQN requires to beat the human score (Mnih et al., 2015)(" $-$ " means BDQN could not beat human score), and SC^+: the number of interactions the BDQN requires to beat the score of DDQN$^+$. Note that Table 3.2 does not aim to compare different methods. Additionally, there are many additional details that are not included in the mentioned papers which can significantly change the algorithms behaviors (Henderson et al., 2017)), e.g., the reported scores of DDQN in Osband et al. (2016) are significantly higher than the reported scores in the original DDQN paper, indicating many existing non-addressed advancements (in the Appendix of (Azizzadenesheli and Anandkumar, 2018)).

We also implemented DDQN drop-out a Thomson Sampling based algorithm motivated by Gal and Ghahramani (2016). We observed that it is not capable of capturing the statistical uncertainty in the Q function and falls short in outperforming a random(uniform) policy. Osband et al. (2016) investigates the sufficiency of the estimated uncertainty and hardness in driving suitable exploitation out of it. It has been observed that drop-out results in the ensemble of infinitely many models but all models almost the same (Dhillon et al., 2018; Osband et al., 2016) in the Appendix of (Azizzadenesheli and Anandkumar, 2018).

As mentioned before, due to an efficient exploration-exploitation strategy, not only BDQN

improves the regret and enhance the sample complexity, but also reaches significantly higher scores. In contrast to naive exploration, BDQN assigns less priority to explore actions that are already observed to be not worthy, resulting in *better sample complexity*. Moreover, since BDQN does not commit to adverse actions, it does not waste the model capacity to estimate the value of unnecessary actions in unnecessary states as good as the important ones, resulting in *saving the model capacity* and better policies.

For the game *Atlantis*, DDQN$^+$ reaches score of $64.67k$ during the evaluation phase, while BDQN reaches score of $3.24M$ after $20M$ time steps. After multiple run of BDQN, we constantly observed that its performance suddenly improves to around $3M$ in the vicinity of $20M$ time steps. We closely investigate this behaviour and realized that BDQN saturates the *Atlantis* game and reaches reaches the internal *OpenAIGym* limit of *max_episode*. After removing this limit, BDQN reaches score $62M$ after $15M$. Please refer to the Appendix of (Azizzadenesheli and Anandkumar, 2018) for the extensive empirical study.

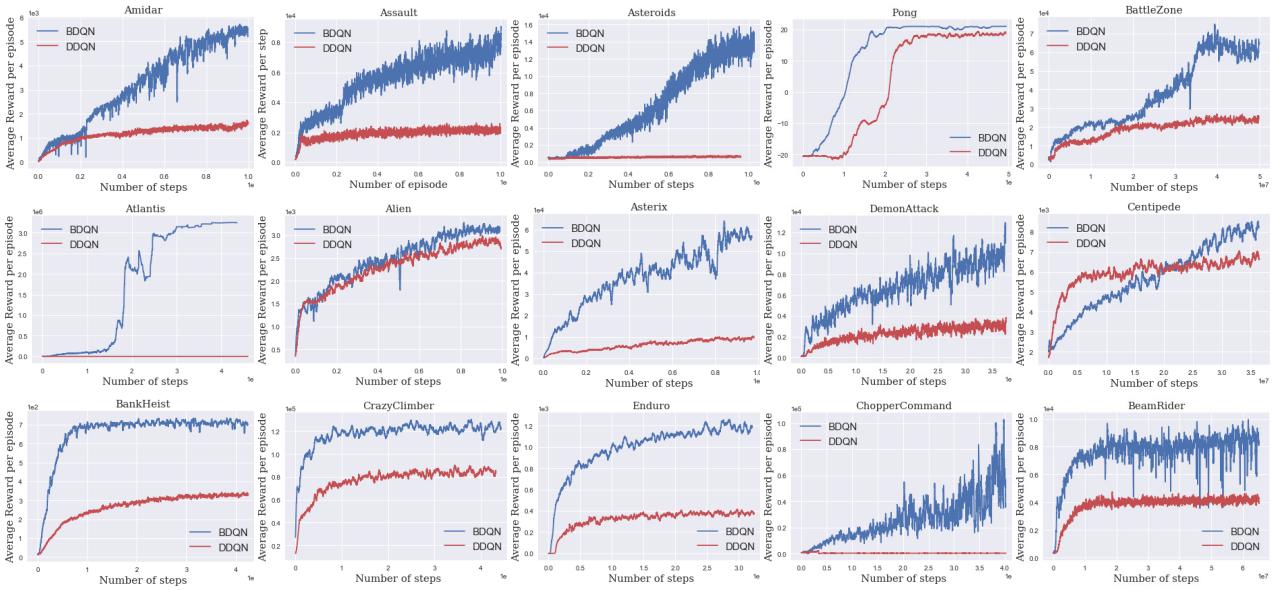


Figure 3.2: The comparison between DDQN and BDQN

Table 3.2: Comparison of scores and sample complexities (scores in the first two columns are average of 100 consecutive episodes). The scores of DDQN$^+$ are the reported scores of DDQN in Van Hasselt et al. (2016) after running it for 200M interactions at evaluation time where the $\varepsilon = 0.001$. Bootstrap DQN (Osband et al., 2016), CTS, Pixel, Reactor (Ostrovski et al., 2017) are borrowed from the original papers. For NoisyNet (Fortunato et al., 2017), the scores of NoisyDQN are reported. Sample complexity, SC: the number of samples the BDQN requires to beat the human score (Mnih et al., 2015)(" $-$ " means BDQN could not beat human score). SC^+: the number of interactions the BDQN requires to beat the score of DDQN$^+$.

Game	BDQN	DDQN	DDQN$^+$	Bootstrap	NoisyNet	CTS	Pixel	Reactor	Human	SC	SC^+	Step
Amidar	**5.52k**	0.99k	0.7k	1.27k	1.5k	1.03k	0.62k	1.18k	1.7k	22.9M	4.4M	100M
Alien	3k	2.9k	2.9k	2.44k	2.9k	1.9k	1.7k	**3.5k**	6.9k	-	36.27M	100M
Assault	**8.84k**	2.23k	5.02k	8.05k	3.1k	2.88k	1.25k	3.5k	1.5k	1.6M	24.3M	100M
Asteroids	**14.1k**	0.56k	0.93k	1.03k	2.1k	3.95k	0.9k	1.75k	13.1k	58.2M	9.7M	100M
Asterix	**58.4k**	11k	15.15k	19.7k	11.0	9.55k	1.4k	6.2k	8.5k	3.6M	5.7M	100M
BeamRider	8.7k	4.2k	7.6k	**23.4k**	14.7k	7.0k	3k	3.8k	5.8k	4.0M	8.1M	70M
BattleZone	**65.2k**	23.2k	24.7k	36.7k	11.9k	7.97k	10k	45k	38k	25.1M	14.9M	50M
Atlantis	3.24M	39.7k	64.76k	99.4k	7.9k	1.8M	40k	**9.5M**	29k	3.3M	5.1M	40M
DemonAttack	11.1k	3.8k	9.7k	82.6k	26.7k	**39.3k**	1.3k	7k	3.4k	2.0M	19.9M	40M
Centipede	**7.3k**	6.4k	4.1k	4.55k	3.35k	5.4k	1.8k	3.5k	12k	-	4.2M	40M
BankHeist	0.72k	0.34k	0.72k	**1.21k**	0.64k	1.3k	0.42k	1.1k	0.72k	2.1M	10.1M	40M
CrazyClimber	124k	84k	102k	**138k**	121k	112.9k	75k	119k	35.4k	0.12M	2.1M	40M
ChopperCmd	**72.5k**	0.5k	4.6k	4.1k	5.3k	5.1k	2.5k	4.8k	9.9k	4.4M	2.2M	40M
Enduro	1.12k	0.38k	0.32k	1.59k	0.91k	0.69k	0.19k	**2.49k**	0.31k	0.82M	0.8M	30M
Pong	**21**	18.82	**21**	20.9	**21**	20.8	17	20	9.3	1.2M	2.4M	5M

3.5 Related Work

The complexity of the exploration-exploitation trade-off has been deeply investigated in RL literature for both continuous and discrete MDPs (Kearns and Singh, 2002; Brafman and Tennenholtz, 2003; Asmuth et al., 2009; Kakade et al., 2003; Ortner and Ryabko, 2012; Osband and Van Roy, 2014a,b). Jaksch et al. (2010a) investigate the regret analysis of MDPs with finite state and action and deploy OFU (Auer, 2003) to guarantee a regret upper bound, while Ortner and Ryabko (2012) relaxes it to a continuous state space and propose a sub-linear regret bound. Azizzadenesheli et al. (2016c) deploys OFU and propose a regret upper bound for Partially Observable MDPs (POMDPs) using spectral methods (Anandkumar et al., 2014). Furthermore, Bartók et al. (2014) tackles a general case of partial monitoring games and provides minimax regret guarantee. For linear quadratic models OFU is deployed

to provide an optimal regret bound (Abbasi-Yadkori and Szepesvári, 2011). In multi-arm bandit, Thompson sampling has been studied both from empirical and theoretical point of views (Chapelle and Li, 2011; Agrawal and Goyal, 2012; Russo and Van Roy, 2014a). A natural adaptation of this algorithm to RL, posterior sampling RL (PSRL) Strens (2000) also shown to have good frequentist and Bayesian performance guarantees (Osband et al., 2013; Abbasi-Yadkori and Szepesvári, 2015). Inevitably for PSRL, these methods also have hard time to become scalable to high dimensional problems, (Ghavamzadeh et al., 2015; Engel et al., 2003; Dearden et al., 1998; Tziortziotis et al., 2013).

Exploration-exploitation trade-offs has been theoretically studied in RL but a prominent problem in high dimensional environments (Mnih et al., 2015; Abel et al., 2016; Azizzadenesheli et al., 2016b). Recent success of Deep RL on Atari games (Mnih et al., 2015), the board game Go (Silver et al., 2017), robotics (Levine et al., 2016), self-driving cars (Shalev-Shwartz et al., 2016), and safety in RL (Lipton et al., 2016a) propose promises on deploying deep RL in high dimensional problem.

To extend the exploration-exploitation efficient methods to high dimensional RL problems, Osband et al. (2016) suggests bootstrapped-ensemble approach that trains several models in parallel to approximate the posterior distribution. Bellemare et al. (2016) propose a way to come up with a surrogate to optimism in high dimensional RL. Other works suggest using a variational approximation to the Q-networks (Lipton et al., 2016b) or a concurrent work on noisy network (Fortunato et al., 2017) suggest to randomize the Q-network. However, most of these approaches significantly increase the computational cost of DQN, e.g., the bootstrapped-ensemble incurs a computation overhead that is linear in the number of bootstrap models.

Concurrently, Levine et al. (2017) proposes least-squares temporal difference which learns a linear model on the feature representation in order to estimate the Q-function. They use ε-greedy approach and provide results on five Atari games. Out of these five games, one is

common with our set of 15 games which BDQN outperforms it by a factor of 360% (w.r.t. the score reported in their paper). As also suggested by our theoretical derivation, our empirical study illustrates that performing Bayesian regression instead, and sampling from the result yields a substantial benefit. This indicates that it is not just the higher data efficiency at the last layer, but that leveraging an explicit uncertainty representation over the value function is of substantial benefit.

3.6 Conclusion

In this work, we proposed LinPSRL and LinUCB, two LinReL algorithms for continuous MDPs. We then proposed BDQN, a deep RL extension of these methods to high dimensional environments. BDQN deploys Thompson sampling and provides an efficient exploration/exploitation in a computationally efficient manner. It involved making simple modifications to the DDQN architecture by replacing the linear regression learning of the last layer with Bayesian linear regression. We demonstrated significantly improvement training, convergence, and regret along with much better performance in many games.

While our current regret upper bounds seem to be sub-optimal in terms of H (we are not aware of any tight lower bound), in the future, we plan to deploy the analysis in (Antos et al., 2008; Lazaric et al., 2010) and develop a tighter regret upper bounds as well as an information theoretic lower bound. We also plan to extend the analysis in Abeille and Lazaric (2017) and develop Thompson sampling methods with a performance guarantee and finally go beyond the linear models (Jiang et al., 2016). While finding optimal continuous action given a Q function can be computationally intractable, we aim to study the relaxation of these approaches in continuous control tasks in the future.

Chapter 4

Safe RL

Combating Reinforcement Learning's Sisyphean Curse with Intrinsic Fear

Many practical environments contain catastrophic states that an optimal agent would visit infrequently or never. Even on toy problems, Deep Reinforcement Learning (DRL) agents tend to periodically revisit these states upon forgetting their existence under a new policy. We introduce *intrinsic fear* (IF), a learned reward shaping that guards DRL agents against periodic catastrophes. IF agents possess a *fear* model trained to predict the probability of imminent catastrophe. This score is then used to penalize the Q-learning objective. Our theoretical analysis bounds the reduction in average return due to learning on the perturbed objective. We also prove robustness to classification errors. As a bonus, IF models tend to learn faster, owing to reward shaping. Experiments demonstrate that *intrinsic-fear* DQNs solve otherwise pathological environments and improve on several Atari games.

4.1 Introduction

Following the success of deep reinforcement learning (DRL) on Atari games Mnih et al. (2015) and the board game of Go Silver et al. (2017), researchers are increasingly exploring practical applications. Some investigated applications include robotics Levine et al. (2016), dialogue systems Fatemi et al. (2016); Lipton et al. (2016b), energy management Night (2016), and self-driving cars Shalev-Shwartz et al. (2016). Amid this push to apply DRL, we might ask, *can we trust these agents in the wild?* Agents acting society may cause harm. A self-driving car might hit pedestrians and a domestic robot might injure a child. Agents might also cause self-injury, and while Atari lives lost are inconsequential, robots are expensive.

Unfortunately, it may not be feasible to prevent all catastrophes without requiring extensive prior knowledge Garcıa and Fernández (2015). Moreover, for typical DQNs, providing large negative rewards does not solve the problem: as soon as the catastrophic trajectories are flushed from the replay buffer, the updated Q-function ceases to discourage revisiting these states.

In this paper, we define *avoidable catastrophes* as states that prior knowledge dictates an optimal policy should visit rarely or never. Additionally, we define *danger states*—those from which a catastrophic state can be reached in a small number of steps, and assume that the optimal policy does visit the danger states rarely or never. The notion of a danger state might seem odd absent any assumptions about the transition function. With a fully-connected transition matrix, all states are danger states. However, physical environments are not fully connected. A car cannot be parked this second, underwater one second later.

This work primarily addresses how we might prevent DRL agents from perpetually making the same mistakes. As a bonus, we show that the prior knowledge knowledge that *catastrophic states* should be avoided accelerates learning. Our experiments show that even on simple toy problems, the classic deep Q-network (DQN) algorithm fails badly, repeatedly

visiting catastrophic states so long as they continue to learn. This poses a formidable obstacle to using DQNs in the real world. How can we trust a DRL-based agent that was doomed to periodically experience catastrophes, just to remember that they exist? Imagine a self-driving car that had to periodically hit a few pedestrians to remember that it is undesirable.

In the tabular setting, an RL agent never forgets the learned dynamics of its environment, even as its policy evolves. Moreover, when the Markovian assumption holds, convergence to a globally optimal policy is guaranteed. However, the tabular approach becomes infeasible in high-dimensional, continuous state spaces. The trouble for DQNs owes to the use of function approximation Murata and Ozawa (2005). When training a DQN, we successively update a neural network based on experiences. These experiences might be sampled in an online fashion, from a trailing window (*experience replay buffer*), or uniformly from all past experiences. Regardless of which mode we use to train the network, eventually, states that a learned policy never encounters will come to form an infinitesimally small region of the training distribution. At such times, our networks suffer the well-known problem of catastrophic forgetting McCloskey and Cohen (1989); McClelland et al. (1995). Nothing prevents the DQN's policy from drifting back towards one that revisits forgotten catastrophic mistakes.

We illustrate the brittleness of modern DRL algorithms with a simple pathological problem called *Adventure Seeker*. This problem consists of a one-dimensional continuous state, two actions, simple dynamics, and admits an analytic solution. Nevertheless, the DQN fails. We then show that similar dynamics exist in the classic RL environment Cart-Pole.

To combat these problems, we propose the *intrinsic fear* (IF) algorithm. In this approach, we train a supervised *fear model* that predicts which states are likely to lead to a catastrophe within k_r steps. The output of the fear model (a probability), scaled by a *fear factor* penalizes the Q-learning target. Crucially, the fear model maintains buffers of both *safe* and *danger*

states. This model never forgets danger states, which is possible due to the infrequency of catastrophes.

We validate the approach both empirically and theoretically. Our experiments address *Adventure Seeker*, Cartpole, and several Atari games. In these environments, we label every lost *life* as a catastrophe. On the toy environments, IF agents learns to avoid catastrophe indefinitely. In Seaquest experiments, the IF agent achieves higher reward and in Asteroids, the IF agent achieves both higher reward and fewer catastrophes. The improvement on Freeway is most dramatic.

We also make the following theoretical contributions: First, we prove that when the reward is bounded and the optimal policy rarely visits the danger states, an optimal policy learned on the perturbed reward function has approximately the same return as the optimal policy learned on the original value function. Second, we prove that our method is robust to noise in the danger model.

4.2 Intrinsic fear

An agent interacts with its environment via a Markov decision process, or MDP, $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$. At each step t, the agent observes a state $s \in \mathcal{S}$ and then chooses an action $a \in \mathcal{A}$ according to its policy π. The environment then transitions to state $s_{t+1} \in \mathcal{S}$ according to transition dynamics $\mathcal{T}(s_{t+1}|s_t, a_t)$ and generates a reward r_t with expectation $\mathcal{R}(s, a)$. This cycle continues until each episode terminates.

An agent seeks to maximize the cumulative discounted return $\sum_{t=0}^{T} \gamma^t r_t$. Temporal-differences methods Sutton (1988) like Q-learning Watkins and Dayan (1992a) model the Q-function, which gives the *optimal* discounted total reward of a state-action pair. Problems of practical interest tend to have large state spaces, thus the Q-function is typically approximated by

parametric models such as neural networks.

In Q-learning with function approximation, an agent collects experiences by acting greedily with respect to $Q(s, a; \theta_Q)$ and updates its parameters θ_Q. Updates proceed as follows. For a given experience (s_t, a_t, r_t, s_{t+1}), we minimize the squared Bellman error:

$$\mathcal{L} = (Q(s_t, a_t; \theta_Q) - y_t)^2 \tag{4.1}$$

for $y_t = r_t + \gamma \cdot \max_{a'} Q(s_{t+1}, a'; \theta_Q)$. Traditionally, the parameterised $Q(s, a; \theta)$ is trained by stochastic approximation, estimating the loss on each experience as it is encountered, yielding the update:

$$\theta_{t+1} \leftarrow \theta_t + \alpha(y_t - Q(s_t, a_t; \theta_t))\nabla Q(s_t, a_t; \theta_t)\,. \tag{4.2}$$

Q-learning methods also require an exploration strategy for action selection. For simplicity, we consider only the ϵ-greedy heuristic. A few tricks help to stabilize Q-learning with function approximation. Notably, with experience replay Lin (1992), the RL agent maintains a buffer of experiences, of experience to update the Q-function.

We propose a new formulation: Suppose there exists a subset $\mathcal{C} \subset \mathcal{S}$ of known *catastrophe states*/ And assume that for a given environment, the optimal policy rarely enters from which catastrophe states are reachable in a short number of steps. We define the distance $d(s_i, s_j)$ to be length N of the smallest sequence of transitions $\{(s_t, a_t, r_t, s_{t+1})\}_{t=1}^{N}$ that traverses state space from s_i to s_j.[1]

Definition 4.1. *Suppose a priori knowledge that acting according to the optimal policy π^*, an agent rarely encounters states $s \in S$ that lie within distance $d(s, c) < k_\tau$ for any catastrophe state $c \in \mathcal{C}$. Then each state s for which $\exists c \in \mathcal{C}$ s.t. $d(s, c) < k_\tau$ is a* danger state.

[1]In the stochastic dynamics setting, the distance is the minimum mean passing time between the states.

Algorithm 5 Training DQN with Intrinsic Fear

Input: Q (DQN), F (fear model), fear factor λ, fear phase-in length k_λ, fear radius k_r
Output: Learned parameters θ_Q and θ_F
Initialize parameters θ_Q and θ_F randomly
Initialize replay buffer $\mathcal{D}$, danger state buffer $\mathcal{D}_D$, and safe state buffer $\mathcal{D}_S$
Start per-episode turn counter n_e
for t in 1:T **do**
  With probability ϵ select random action a_t
  Otherwise, select a greedy action $a_t = \arg\max_a Q(s_t, a; \theta_Q)$
  Execute action a_t in environment, observing reward r_t and successor state s_{t+1}
  Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
  if s_{t+1} is a catastrophe state **then**
    Add states s_{t-k_r} through s_t to $\mathcal{D}_D$
  else
    Add states s_{t-n_e} through s_{t-k_r-1} to $\mathcal{D}_S$
  end if
  Sample a random mini-batch of transitions $(s_\tau\ ,\ a_\tau,\ r_\tau,\ s_{\tau+1})$ from $\mathcal{D}$
  $\lambda_\tau \leftarrow \min(\lambda, \frac{\lambda \cdot t}{k_\lambda})$
  $y_\tau \leftarrow \left\{ \begin{array}{ll} \text{for terminal } s_{\tau+1}: & r_\tau - \lambda_\tau \\ \text{for non-terminal } s_{\tau+1}: & r_\tau + \max_{a'} Q(s_{\tau+1}, a'; \theta_Q) - \lambda \cdot F(s_{\tau+1}; \theta_F) \end{array} \right\}$
  $\theta_Q \leftarrow \theta_Q - \eta \cdot \nabla_{\theta_Q}(y_\tau - Q(s_\tau, a_\tau; \theta_Q))^2$
  Sample random mini-batch s_j with 50% of examples from $\mathcal{D}_D$ and 50% from $\mathcal{D}_S$
  $y_j \leftarrow \left\{ \begin{array}{ll} 1, & \text{for } s_j \in \mathcal{D}_D \\ 0, & \text{for } s_j \in \mathcal{D}_S \end{array} \right\}$
  $\theta_F \leftarrow \theta_F - \eta \cdot \nabla_{\theta_F} \text{loss}_F(y_j, F(s_j; \theta_F))$
end for

In Algorithm 5, the agent maintains both a DQN and a separate, supervised *fear model* $F : \mathcal{S} \mapsto [0, 1]$. F provides an auxiliary source of reward, penalizing the Q-learner for entering likely danger states. In our case, we use a neural network of the same architecture as the DQN (but for the output layer). While one could sharing weights between the two networks, such tricks are not relevant to this paper's contribution.

We train the fear model to predict the probability that any state will lead to catastrophe within k moves. Over the course of training, our agent adds each experience (s, a, r, s') to its experience replay buffer. Whenever a catastrophe is reached at, say, the n_{th} turn of an episode, we add the preceding k_r (*fear radius*) states to a *danger buffer*. We add the first $n - k_r$ states of that episode to a *safe buffer*. When $n < k_r$, all states for that episode are added to the list of danger states. Then after each turn, in addition to updating the Q-network, we update the fear model, sampling 50% of states from the *danger buffer*, assigning them label 1, and the remaining 50% from the *safe buffer*, assigning them label 0.

For each update to the DQN, we perturb the TD target y_t. Instead of updating $Q(s_t, a_t; \theta_Q)$ towards $r_t + \max_{a'} Q(s_{t+1}, a'; \theta_Q)$, we modify the target by subtracting the *intrinsic fear*:

$$y_t^{IF} = r_t + \max_{a'} Q(s_{t+1}, a'; \theta_Q) - \lambda \cdot F(s_{t+1}; \theta_F) \tag{4.3}$$

where $F(s; \theta_F)$ is the fear model and λ is a *fear factor* determining the scale of the impact of intrinsic fear on the Q-function update.

4.3 Analysis

Note that IF perturbs the objective function. Thus, one might be concerned that the perturbed reward might lead to a sub-optimal policy. Fortunately, as we will show formally, if the labeled catastrophe states and danger zone do not violate our assumptions, and if the

fear model reaches arbitrarily high accuracy, then this will not happen.

For an MDP, $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, with $0 \leq \gamma \leq 1$, the average reward return is as follows:

$$\eta_M(\pi) := \begin{cases} \lim_{T\to\infty} \frac{1}{T}\mathbb{E}_M\Big[\sum_t^T r_t|\pi\Big] & \textit{if} \quad \gamma = 1 \\ (1-\gamma)\mathbb{E}_M\Big[\sum_t^\infty \gamma^t r_t|\pi\Big] & \textit{if} \quad 0 \leq \gamma < 1 \end{cases}$$

The optimal policy π^* of the model M is the policy which maximizes the average reward return, $\pi^* = \max_{\pi\in\mathcal{P}} \eta(\pi)$ where $\mathcal{P}$ is a set of stationary polices.

Theorem 4.1. *For a given MDP, M, with $\gamma \in [0,1]$ and a catastrophe detector f, let π^* denote* any *optimal policy of M, and $\tilde{\pi}$ denote an optimal policy of M equipped with fear model F, and λ, environment (M, F). If the probability π^* visits the states in the danger zone is at most ϵ, and $0 \leq \mathcal{R}(s,a) \leq 1$, then*

$$\eta_M(\pi^*) \geq \eta_M(\tilde{\pi}) \geq \eta_{M,F}(\tilde{\pi}) \geq \eta_M(\pi^*) - \lambda\epsilon\,. \tag{4.4}$$

In other words, $\tilde{\pi}$ is $\lambda\epsilon$-optimal in the original MDP.

Proof. The policy π^* visits the fear zone with probability at most ϵ. Therefore, applying π^* on the environment with intrinsic fear (M, F), provides a expected return of at least $\eta_M(\pi^*) - \epsilon\lambda$. Since there exists a policy with this expected return on (M, F), therefore, the optimal policy of (M, F), must result in an expected return of at least $\eta_M(\pi^*) - \epsilon\lambda$ on (M, F), i.e. $\eta_{M,F}(\tilde{\pi}) \geq \eta_M(\pi^*) - \epsilon\lambda$. The expected return $\eta_{M,F}(\tilde{\pi})$ decomposes into two parts: (i) the expected return from original environment M, $\eta_M(\tilde{\pi})$, (ii) the expected return from the fear model. If $\tilde{\pi}$ visits the fear zone with probability at most $\tilde{\epsilon}$, then $\eta_{M,F}(\tilde{\pi}) \geq \eta_M(\tilde{\pi}) - \lambda\tilde{\epsilon}$. Therefore, applying $\tilde{\pi}$ on M promises an expected return of at least $\eta_M(\pi^*) - \epsilon\lambda + \tilde{\epsilon}\lambda$, lower bounded by $\eta_M(\pi^*) - \epsilon\lambda$. □

It is worth noting that the theorem holds for *any* optimal policy of M. If one of them does not visit the fear zone at all (i.e., $\epsilon = 0$), then $\eta_M(\pi^*) = \eta_{M,F}(\tilde{\pi})$ and the fear signal can boost up the process of learning the optimal policy.

Since we empirically learn the fear model F using collected data of some finite sample size N, our RL agent has access to an imperfect fear model $\hat{F}$, and therefore, computes the optimal policy based on $\hat{F}$. In this case, the RL agent trains with intrinsic fear generated by $\hat{F}$, learning a different value function than the RL agent with perfect F. To show the robustness against errors in $\hat{F}$, we are interested in the average deviation in the value functions of the two agents.

Our second main theoretical result, given in Theorem 4.2, allows the RL agent to use a smaller discount factor, denoted γ_{plan}, than the actual one ($\gamma_{plan} \leq \gamma$), to reduce the planning horizon and computation cost. Moreover, when an estimated model of the environment is used, Jiang et al. (2015) shows that using a smaller discount factor for planning may prevent over-fitting to the estimated model. Our result demonstrates that using a smaller discount factor for planning can reduce reduction of expected return when an estimated fear model is used.

Specifically, for a given environment, with fear model F_1 and discount factor γ_1, let $V_{F_1,\gamma_1}^{\pi^*_{F_2,\gamma_2}}(s),\ s \in \mathcal{S}$, denote the state value function under the optimal policy of an environment with fear model F_2 and the discount factor γ_2. In the same environment, let $\omega^{\pi}(s)$ denote the visitation distribution over states under policy π. We are interested in the average reduction on expected return caused by an imperfect classifier; this reduction, denoted $\mathcal{L}(F, \widehat{F}, \gamma, \gamma_{plan})$, is defined as

$$(1-\gamma)\int_{s\in\mathcal{S}} \omega^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)\left(V_{F,\gamma}^{\pi^*_{F,\gamma}}(s) - V_{F,\gamma}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)\right) ds\,.$$

Theorem 4.2. *Suppose $\gamma_{plan} \leq \gamma$, and $\delta \in (0,1)$. Let $\hat{F}$ be the fear model in $\mathcal{F}$ with*

minimum empirical risk on N samples. For a given MDP model, the average reduction on expected return, $\mathcal{L}(F, \widehat{F}, \gamma, \gamma_{plan})$, vanishes as N increase: with probability at least $1-\delta$,

$$\mathcal{L} = \mathcal{O}\left(\lambda \frac{1-\gamma}{1-\gamma_{plan}} \frac{\mathcal{VC}(\mathcal{F}) + \log\frac{1}{\delta}}{N} + \frac{(\gamma - \gamma_{plan})}{1-\gamma_{plan}}\right), \tag{4.5}$$

where $\mathcal{VC}(\mathcal{F})$ is the $\mathcal{VC}$ dimension of the hypothesis class $\mathcal{F}$.

Proof. In order to analyze $\left(V_{F,\gamma}^{\pi^*_{F,\gamma}}(s) - V_{F,\gamma}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)\right)$, which is always non-negative, we decompose it as follows:

$$\left(V_{F,\gamma}^{\pi^*_{F,\gamma}}(s) - V_{F,\gamma_{plan}}^{\pi^*_{F,\gamma}}(s)\right) + \left(V_{F,\gamma_{plan}}^{\pi^*_{F,\gamma}}(s) - V_{F,\gamma}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)\right) \tag{4.6}$$

The first term is the difference in the expected returns of $\pi^*_{F,\gamma}$ under two different discount factors, starting from s:

$$\mathbb{E}\left[\sum_{t=0}^{\infty}(\gamma^t - \gamma_{plan}^t) r_t | s_0 = s, \pi^*_{F,\gamma}, F, M\right]. \tag{4.7}$$

Since $r_t \leq 1,\ \forall t$, using the geometric series, Eq. 4.7 is upper bounded by $\frac{1}{1-\gamma} - \frac{1}{1-\gamma_{plan}} = \frac{\gamma-\gamma_{plan}}{(1-\gamma_{plan})(1-\gamma)}$.

The second term is upper bounded by $V_{F,\gamma_{plan}}^{\pi^*_{F,\gamma_{plan}}}(s) - V_{F,\gamma}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)$ since $\pi^*_{F,\gamma_{plan}}$ is an optimal policy of an environment equipped with (F, γ_{plan}). Furthermore, as $\gamma_{plan} \leq \gamma$ and $r_t \geq 0$, we have $V_{F,\gamma}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s) \geq V_{F,\gamma_{plan}}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)$. Therefore, the second term of Eq. 4.6 is upper bounded by $V_{F,\gamma_{plan}}^{\pi^*_{F,\gamma_{plan}}}(s) - V_{F,\gamma_{plan}}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)$, which is the deviation of the value function under two different close policies. Since F and $\widehat{F}$ are close, we expect that this deviation to be small. With one

more decomposition step

$$
\begin{aligned}
&V_{F,\gamma_{plan}}^{\pi^*_{F,\gamma_{plan}}}(s) - V_{F,\gamma_{plan}}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s) \\
&\quad = \left(V_{F,\gamma_{plan}}^{\pi^*_{F,\gamma_{plan}}}(s) - V_{\widehat{F},\gamma_{plan}}^{\pi^*_{F,\gamma_{plan}}}(s)\right) + \left(V_{\widehat{F},\gamma_{plan}}^{\pi^*_{F,\gamma_{plan}}}(s) - V_{\widehat{F},\gamma_{plan}}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)\right) + \left(V_{\widehat{F},\gamma_{plan}}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s) - V_{F,\gamma_{plan}}^{\pi^*_{\widehat{F},\gamma_{plan}}}(s)\right)
\end{aligned}
$$

Since the middle term in this equation is non-positive, we can ignore it for the purpose of upper-bounding the left-hand side. The upper bound is sum of the remaining two terms which is also upper bounded by 2 times of the maximum of them;

$$
2 \max_{\pi \in \{\pi^*_{F,\gamma_{plan}}, \pi^*_{\widehat{F},\gamma_{plan}}\}} \left| V_{\widehat{F},\gamma_{plan}}^{\pi}(s) - V_{F,\gamma_{plan}}^{\pi}(s) \right| ,
$$

which is the deviation in values of different domains. The value functions satisfy the Bellman equation for any π:

$$
\begin{aligned}
V_{F,\gamma_{plan}}^{\pi}(s) =& \mathcal{R}(s,\pi(s)) + \lambda F(s) + \gamma_{plan} \int_{s' \in \mathcal{S}} \mathcal{T}(s'|s,\pi(s)) V_{F,\gamma_{plan}}^{\pi}(s') ds \\
V_{\widehat{F},\gamma_{plan}}^{\pi}(s) =& \mathcal{R}(s,\pi(s)) + \lambda \widehat{F}(s) + \gamma_{plan} \int_{s' \in \mathcal{S}} \mathcal{T}(s'|s,\pi(s)) V_{\widehat{F},\gamma_{plan}}^{\pi}(s') ds
\end{aligned} \tag{4.8}
$$

which can be solved using iterative updates of dynamic programing. Let $V_i^{\pi}(s)$ and $\widehat{V}_i^{\pi}(s)$ respectably denote the i'th iteration of the dynamic programmings corresponding to the first and second equalities in Eq. 4.8. Therefore, for any state

$$
\begin{aligned}
&V_i^{\pi}(s) - \widehat{V}_i^{\pi}(s) \\
&\quad = \lambda' F(s) - \lambda' \widehat{F}(s) + \gamma_{plan} \int_{s' \in \mathcal{S}} \mathcal{T}(s'|s,\pi(s)) \left(V_{i-1}(s') - \widehat{V}_{i-1}(s')\right) ds \leq \lambda \sum_{i'=0}^{i} \left(\gamma_{plan} \mathcal{T}^{\pi}\right)^{i'} \left(F - \widehat{F}\right.
\end{aligned} \tag{4.9}
$$

where $(\mathcal{T}^{\pi})^i$ is a kernel and denotes the transition operator applied i times to itself. The classification error $\left|F(s) - \widehat{F}(s)\right|$ is the zero-one loss of binary classifier, therefore, its ex-

pectation $\int_{s\in\mathcal{S}} \omega^{\pi^*_{\widehat{F},\gamma_{plan}}}(s) \left|F(s) - \widehat{F}(s)\right| ds$ is bounded by $3200\frac{\mathcal{VC}(\mathcal{F})+\log\frac{1}{\delta}}{N}$ with probability at least $1-\delta$ Vapnik (2013); Hanneke (2016). As long as the operator $(\mathcal{T}^\pi)^i$ is a linear operator,

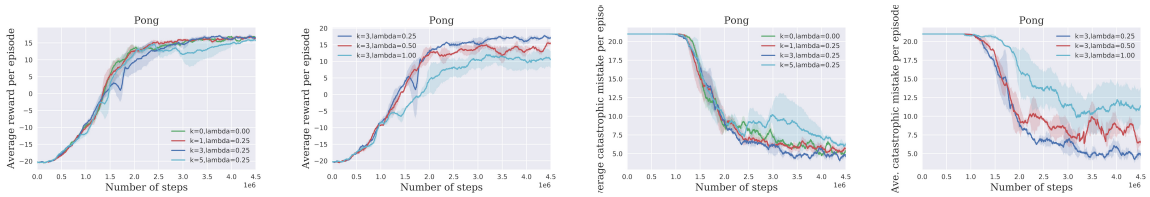
$$\int_{s\in\mathcal{S}} \omega^{\pi^*_{\widehat{F},\gamma_{plan}}}(s) \left|V_i^\pi(s) - \widehat{V}_i^\pi(s)\right| ds \leq \lambda \frac{3200}{1-\gamma_{plan}} \frac{\mathcal{VC}(\mathcal{F}) + \log\frac{1}{\delta}}{N} \,. \tag{4.10}$$

Therefore, $\mathcal{L}(F, \widehat{F}, \gamma, \gamma_{plan})$ is bounded by $(1-\gamma)$ times of sum of Eq. 4.10 and $\frac{1-\gamma}{1-\gamma_{plan}}$, with probability at least $1-\delta$. □

Theorem 4.2 holds for both finite and continuous state-action MDPs. Over the course of our experiments, we discovered the following pattern: Intrinsic fear models are more effective when the *fear radius* k_r is large enough that the model can experience danger states at a safe distance and correct the policy, without experiencing many catastrophes. When the fear radius is too small, the danger probability is only nonzero at states from which catastrophes are inevitable anyway and intrinsic fear seems not to help. We also found that wider fear factors train more stably when phased in over the course of many episodes. So, in all of our experiments we gradually phase in the *fear factor* from 0 to λ reaching full strength at predetermined time step k_λ.

4.4 Experiments

As it has been mentioned in the Thm. 4.1, the degradation of the optimal policy due to the intrinsic fear, potentially, can be characterized with fear penalty λ and ϵ, which is also a function of radius of the fear zone k. In practice, we use function approximation methods in order to find the optimal policy and as it is mentioned before, the function approximation approaches suffer from various sources of biases, e.g. catastrophic forgetting. In th following experimental studies, we expect by deploying the fear model, the DRL algorithm becomes more robust to the catastrophic mistakes but we expect as the k and λ increase, the DRL



(a) (b) (c) (d)

Figure 4.1: The analyses of the effect of radius k of the fear zone, and λ, the penalty assign to fear zone for the game Pong. 4.1a: The average reward per episode for different radius $k = \{1, 3, 5\}$ and $\lambda = 0.25$ and 4.1a, the corresponding average catastrophic mistakes. 4.1c: The average reward per episode for different $\lambda = \{0.25, 0.50, 1.00\}$ for fixed $k = 3$ and 4.1d, the corresponding average catastrophic mistakes.

behavior degrades. In this section, we study the sensitivity of the DQN model to the choice of these parameters. We vary the radius of the fear zone k in range of $k = \{1, 3, 5\}$ and the fear intrinsic penalty λ in the range of $\{0.25, 0.50, 1.00\}$. We study the DQN behavior accompanied with fear model for four ALE environments, Pong, Chopper Command, Seaquest, and Demon Attack. For each experiment, and a specific configuration of k and λ, we run the experiment for three times and report the mean and variance of the return as well as the occurrence rate of catastrophic mistakes. Each run of the game Pong is $20M$ frame ($5M$ decision steps) and for the rest of the games, each run consist of $80M$ frames ($20M$ decision steps). The plotted returns are through moving average with window if length 100 over episode. For the rate of catastrophic mistakes, for the game Pong, we report the average number of catastrophic mistake per 100 episode Fig. 4.1.

This measure can not capture the effect of inartistic fear model for the games where avoiding the catastrophically event make the length of each episode significantly long or the number of possible catastrophic mistake is constant (the number of life assign to each round of game.). For these type of games, we report the average number of catastrophic mistake divided by the average length of the episodes Fig. 4.2. As it is shown in the Fig. 4.2 when the deployed radius k or the penalty λ increase, the DRL performance develops, while the rate of catastrophic mistakes goes down. When these two parameters increase dramatically, there is the degration in the agent behaviour, .e.g the the agent in Chopper-Command for $k = 3$

and $\lambda = 1.00$ learns a too conservative policy such that the agent does not proceed forward in order to collect rewards https://youtu.be/em-FQMH8mMQ, or Seaquest agent learns to a deadlock move in order to minimize the negative penalties https://youtu.be/dOIqv0afnNE.

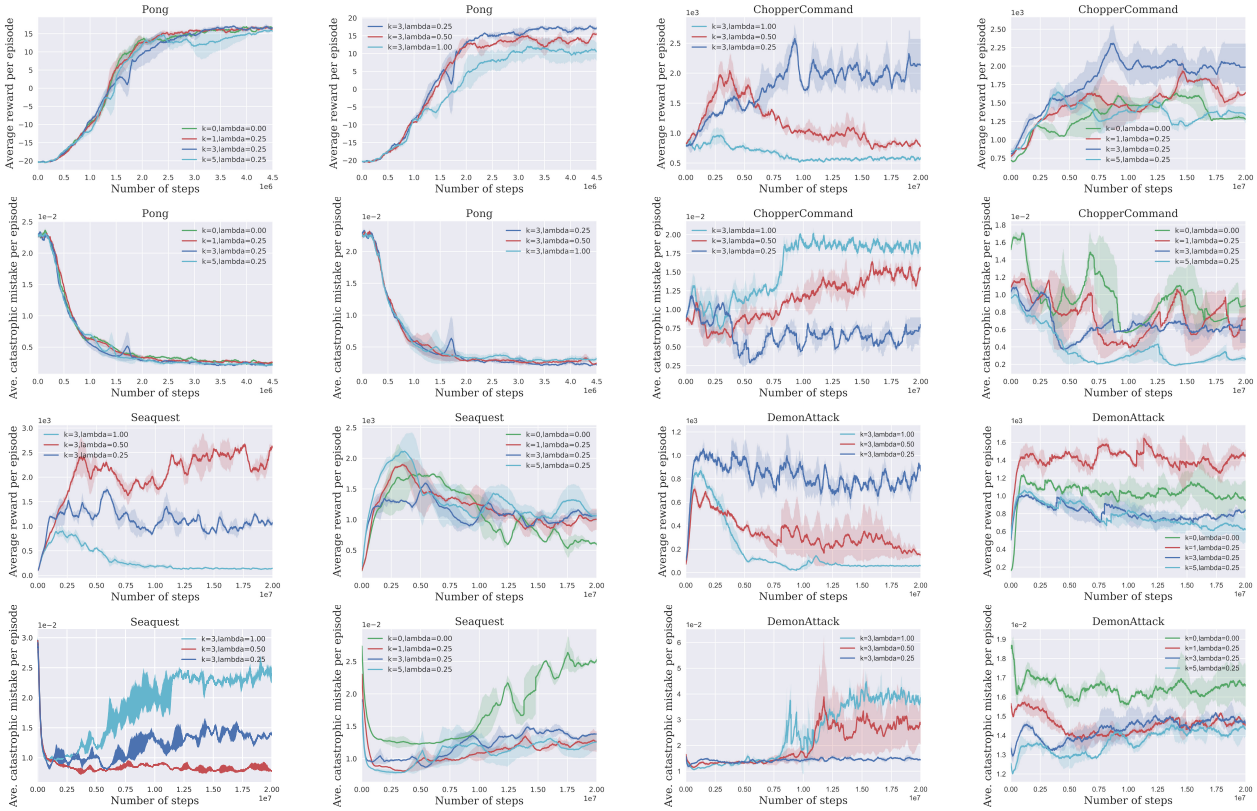


Figure 4.2: The analyses of the effect of radius k of the fear zone, and λ, the penalty assign to fear zone for a set of different games

4.5 Related work

The paper studies safety in RL, intrinsically motivated RL, and the stability of Q-learning with function approximation under distributional shift. Our work also has some connection to reward shaping. We attempt to highlight the most relevant papers here. Several papers address safety in RL. Garcıa and Fernández (2015) provide a thorough review on the topic, identifying two main classes of methods: those that perturb the objective function and those that use external knowledge to improve the safety of exploration.

While a typical reinforcement learner optimizes expected return, some papers suggest that a safely acting agent should also minimize risk. Hans et al. (2008) defines a *fatality* as any return below some threshold τ. They propose a solution comprised of a *safety function*, which identifies unsafe states, and a *backup model*, which navigates away from those states. Their work, which only addresses the tabular setting, suggests that an agent should minimize the probability of fatality instead of maximizing the expected return. Heger (1994) suggests an alternative Q-learning objective concerned with the minimum (vs. expected) return. Other papers suggest modifying the objective to penalize policies with high-variance returns Garcıa and Fernández (2015); Chow et al. (2015). Maximizing expected returns while minimizing their variance is a classic problem in finance, where a common objective is the ratio of expected return to its standard deviation Sharpe (1966). Moldovan and Abbeel (2012) give a definition of safety based on ergodicity. They consider a fatality to be a state from which one cannot return to the start state. Shalev-Shwartz et al. (2016) theoretically analyzes how strong a penalty should be to discourage accidents. They also consider hard constraints to ensure safety. None of the above works address the case where distributional shift dooms an agent to perpetually revisit known catastrophic failure modes. Other papers incorporate external knowledge into the exploration process. Typically, this requires access to an oracle or extensive prior knowledge of the environment. In the extreme case, some papers suggest confining the policy search to a known subset of *safe* policies. For reasonably complex environments or classes of policies, this seems infeasible.

The potential oscillatory or divergent behavior of Q-learners with function approximation has been previously identified Boyan and Moore (1995); Baird (1995); Gordon (1996). Outside of RL, the problem of covariate shift has been extensively studied Sugiyama and Kawanabe (2012). Murata and Ozawa (2005) addresses the problem of catastrophic forgetting owing to distributional shift in RL with function approximation, proposing a memory-based solution. Many papers address intrinsic rewards, which are internally assigned, vs the standard (extrinsic) reward. Typically, intrinsic rewards are used to encourage exploration Schmid-

huber (1991); Bellemare et al. (2016) and to acquire a modular set of skills Chentanez et al. (2004). Some papers refer to the intrinsic reward for discovery as *curiosity*. Like classic work on intrinsic motivation, our methods perturb the reward function. But instead of assigning bonuses to encourage discovery of novel transitions, we assign penalties to discourage catastrophic transitions.

Key differences In this paper, we undertake a novel treatment of safe reinforcement learning, While the literature offers several notions of safety in reinforcement learning, we see the following problem: Existing safety research that perturbs the reward function requires little foreknowledge, but fundamentally changes the objective globally. On the other hand, processes relying on expert knowledge may presume an unreasonable level of foreknowledge. Moreover, little of the prior work on safe reinforcement learning, to the best of our knowledge, specifically addresses the problem of catastrophic forgetting. This paper proposes a new class of algorithms for avoiding catastrophic states and a theoretical analysis supporting its robustness.

4.6 Conclusions

Our experiments demonstrate that DQNs are susceptible to periodically repeating mistakes, however bad, raising questions about their real-world utility when harm can come of actions. While it is easy to visualize these problems on toy examples, similar dynamics are embedded in more complex domains. Consider a domestic robot acting as a barber. The robot might receive positive feedback for giving a closer shave. This reward encourages closer contact at a steeper angle. Of course, the shape of this reward function belies the catastrophe lurking just past the optimal shave. Similar dynamics might be imagines in a vehicle that is rewarded for traveling faster but could risk an accident with excessive speed. Our results with the

intrinsic fear model suggest that with only a small amount of prior knowledge (the ability to recognize catastrophe states after the fact), we can simultaneously accelerate learning and avoid catastrophic states. This work is a step towards combating DRL's tendency to revisit catastrophic states due to catastrophic forgetting.

Chapter 5

RL in Partially Observable MDPs

Reinforcement Learning of POMDPs using Spectral Methods

We propose a new reinforcement learning algorithm for partially observable Markov decision processes (POMDP) based on spectral decomposition methods. While spectral methods have been previously employed for consistent learning of (passive) latent variable models such as hidden Markov models, POMDPs are more challenging since the learner interacts with the environment and possibly changes the future observations in the process. We devise a learning algorithm running through episodes, in each episode we employ spectral techniques to learn the POMDP parameters from a trajectory generated by a fixed policy. At the end of the episode, an optimization oracle returns the optimal memoryless planning policy which maximizes the expected reward based on the estimated POMDP model. We prove an order-optimal regret bound w.r.t. the optimal memoryless policy and efficient scaling with respect to the dimensionality of observation and action spaces.

5.1 Introduction

Reinforcement Learning (RL) is an effective approach to solve the problem of sequential decision–making under uncertainty. RL agents learn how to maximize long-term reward using the experience obtained by direct interaction with a stochastic environment (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998). Since the environment is initially unknown, the agent has to balance between *exploring* the environment to estimate its structure, and *exploiting* the estimates to compute a policy that maximizes the long-term reward. As a result, designing a RL algorithm requires three different elements: **1)** an estimator for the environment's structure, **2)** a planning algorithm to compute the optimal policy of the estimated environment (LaValle, 2006), and **3)** a strategy to make a trade off between exploration and exploitation to minimize the *regret*, i.e., the difference between the performance of the exact optimal policy and the rewards accumulated by the agent over time.

Most of RL literature assumes that the environment can be modeled as a Markov decision process (MDP), with a Markovian state evolution that is fully observed. A number of exploration–exploitation strategies have been shown to have strong performance guarantees for MDPs, either in terms of regret or sample complexity (see Sect. 5.1.2 for a review). For the large state space MDP, where the classical approaches are not scalable, Kocsis and Szepesvári (2006) introduces MDP Monte-Carlo planning tree which is one of the few viable approaches to find the near-optimal policy. However, the assumption of full observability of the state evolution is often violated in practice, and the agent may only have noisy observations of the true state of the environment (e.g., noisy sensors in robotics). In this case, it is more appropriate to use the partially-observable MDP or POMDP (Sondik, 1971) model.

Many challenges arise in designing RL algorithms for POMDPs. Unlike in MDPs, the estimation problem (element 1) involves identifying the parameters of a latent variable model (LVM). In a MDP the agent directly observes (stochastic) state transitions, and the estima-

tion of the generative model is straightforward via empirical estimators. On the other hand, in a POMDP the transition and reward models must be inferred from noisy observations and the Markovian state evolution is hidden. The planning problem (element 2), i.e., computing the optimal policy for a POMDP with known parameters, is PSPACE-complete (Papadimitriou and Tsitsiklis, 1987a), and it requires solving an augmented MDP built on a continuous belief space (i.e., a distribution over the hidden state of the POMDP). Finally, integrating estimation and planning in an exploration–exploitation strategy (element 3) with guarantees is non-trivial and no no-regret strategies are currently known (see Sect. 5.1.2).

5.1.1 Summary of Results

The main contributions of this paper are as follows: (i) We propose a new RL algorithm for POMDPs that incorporates spectral parameter estimation within a exploration-exploitation framework, (ii) we analyze regret bounds assuming access to an optimization oracle that provides the best memoryless planning policy at the end of each learning episode, (iii) we prove order optimal regret and efficient scaling with dimensions, thereby providing the first guaranteed RL algorithm for a wide class of POMDPs.

The estimation of the POMDP is carried out via spectral methods which involve decomposition of certain moment tensors computed from data. This learning algorithm is interleaved with the optimization of the planning policy using an exploration–exploitation strategy inspired by the UCRL method for MDPs (Ortner and Auer, 2007; Jaksch et al., 2010b). The resulting algorithm, called SM-UCRL (*Spectral Method for Upper-Confidence Reinforcement Learning*), runs through episodes of variable length, where the agent follows a fixed policy until enough data are collected and then it updates the current policy according to the estimates of the POMDP parameters and their accuracy. Throughout the paper we focus on the estimation and exploration–exploitation aspects of the algorithm, while we assume

access to a *planning oracle* for the class of memoryless policies (i.e., policies directly mapping observations to a distribution over actions).[1]

Theoretical Results. We prove the following learning result. For the full details see Thm. 5.1 in Sect. 5.3.

Theorem 1. ***(Informal Result on Learning POMDP Parameters)*** *Let M be a POMDP with X states, Y observations, A actions, R rewards, and $Y > X$, and characterized by densities $f_T(x'|x,a)$, $f_O(y|x)$, and $f_R(r|x,a)$ defining state transition, observation, and the reward models. Given a sequence of observations, actions, and rewards generated by executing a memoryless policy where each action a is chosen $N(a)$ times, there exists a spectral method which returns estimates $\widehat{f}_T$, $\widehat{f}_O$, and $\widehat{f}_R$ that, under suitable assumptions on the POMDP, the policy, and the number of samples, satisfy*

$$\|\widehat{f}_O(\cdot|x) - f_O(\cdot|x)\|_1 \leq \widetilde{O}\left(\sqrt{\frac{YR}{N(a)}}\right),$$
$$\|\widehat{f}_R(\cdot|x,a) - f_R(\cdot|x,a)\|_1 \leq \widetilde{O}\left(\sqrt{\frac{YR}{N(a)}}\right),$$
$$\|\widehat{f}_T(\cdot|x,a) - f_T(\cdot|x,a)\|_2 \leq \widetilde{O}\left(\sqrt{\frac{YRX^2}{N(a)}}\right),$$

with high probability, for any state x and any action a.

This result shows the consistency of the estimated POMDP parameters and it also provides explicit confidence intervals.

By employing the above learning result in a UCRL framework, we prove the following bound on the regret Reg_N w.r.t. the optimal memoryless policy. For full details see Thm. 5.2 in Sect. 5.4.

[1]This assumption is common in many works in bandit and RL literature (see e.g., Abbasi-Yadkori and Szepesvári (2011) for linear bandit and Chen et al. (2013) in combinatorial bandit), where the focus is on the exploration–exploitation strategy rather than the optimization problem.

Theorem 2. ***(Informal Result on Regret Bounds)*** *Let M be a POMDP with X states, Y observations, A actions, and R rewards, with a diameter D defined as*

$$D := \max_{x,x' \in \mathcal{X}, a,a' \in \mathcal{A}} \min_{\pi} \mathbb{E}\big[\tau(x', a'|x, a; \pi)\big],$$

i.e., the largest mean passage time between any two state-action pairs in the POMDP using a memoryless policy π mapping observations to actions. If SM-UCRL *is run over N steps using the confidence intervals of Thm. 5.1, under suitable assumptions on the POMDP, the space of policies, and the number of samples, we have*

$$Reg_N \leq \widetilde{O}\Big(DX^{3/2}\sqrt{AYRN}\Big),$$

with high probability.

The above result shows that despite the complexity of estimating the POMDP parameters from noisy observations of hidden states, the regret of SM-UCRL is similar to the case of MDPs, where the regret of UCRL scales as $\widetilde{O}(D_{\text{MDP}} X\sqrt{AN})$. The regret is order-optimal, since $\widetilde{O}(\sqrt{N})$ matches the lower bound for MDPs.

Another interesting aspect is that the diameter of the POMDP is a natural extension of the MDP case. While D_{MDP} measures the mean passage time using state–based policies (i.e., a policies mapping *states* to actions), in POMDPs policies cannot be defined over states but rather on observations and this naturally translates into the definition of the diameter D. More details on other problem-dependent terms in the bound are discussed in Sect. 5.4.

The derived regret bound is with respect to the best memoryless (stochastic) policy for the given POMDP. Indeed, for a general POMDP, the optimal policy need not be memoryless. However, finding the optimal policy is uncomputable for infinite horizon regret minimization (Madani, 1998). Instead memoryless policies have shown good performance in practice

(see the Section on related work). Moreover, for the class of so-called *contextual MDP*, a special class of POMDPs, the optimal policy is also memoryless (Krishnamurthy et al., 2016a).

Analysis of the learning algorithm. The learning results in Thm. 5.1 are based on spectral tensor decomposition methods, which have been previously used for consistent estimation of a wide class of LVMs (Anandkumar et al., 2014). This is in contrast with traditional learning methods, such as expectation-maximization (EM) (Dempster et al., 1977), that have no consistency guarantees and may converge to local optimum which is arbitrarily bad.

While spectral methods have been previously employed in sequence modeling such as in HMMs (Anandkumar et al., 2014), by representing it as multiview model, their application to POMDPs is not trivial. In fact, unlike the HMM, the consecutive observations of a POMDP are no longer conditionally independent, when conditioned on the hidden state of middle *view*. This is because the decision (or the action) depends on the observations themselves. By limiting to memoryless policies, we can control the range of this dependence, and by conditioning on the actions, we show that we can obtain conditionally independent *views*. As a result, starting with samples collected along a trajectory generated by a fixed policy, we can construct a multi-view model and use the tensor decomposition method on each action separately, estimate the parameters of the POMDP, and define confidence intervals.

While the proof follows similar steps as in previous works on spectral methods (e.g., HMMs Anandkumar et al., 2014), here we extend concentration inequalities for dependent random variables to matrix valued functions by combining the results of Kontorovich et al. (2008) with the matrix Azuma's inequality of Tropp (2012). This allows us to remove the usual assumption that the samples are generated from the stationary distribution of the current policy. This is particularly important in our case since the policy changes at each episode

and we can avoid discarding the initial samples and waiting until the corresponding Markov chain converged (i.e., the *burn-in* phase).

The condition that the POMDP has more observations than states ($Y > X$) follows from standard non-degeneracy conditions to apply the spectral method. This corresponds to considering POMDPs where the underlying MDP is defined over a few number of states (i.e., a low-dimensional space) that can produce a large number of noisy observations. This is common in applications such as spoken-dialogue systems (Atrash and Pineau, 2006; Png et al., 2012) and medical applications (Hauskrecht and Fraser, 2000). We also show how this assumption can be relaxed and the result can be applied to a wider family of POMDPs.

Analysis of the exploration–exploitation strategy. SM-UCRL applies the popular *optimism-in-face-of-uncertainty* principle[2] to the confidence intervals of the estimated POMDP and compute the optimal policy of the most optimistic POMDP in the admissible set. This *optimistic* choice provides a smooth combination of the exploration encouraged by the confidence intervals (larger confidence intervals favor uniform exploration) and the exploitation of the estimates of the POMDP parameters.

While the algorithmic integration is rather simple, its analysis is not trivial. The spectral method cannot use samples generated from different policies and the length of each episode should be carefully tuned to guarantee that estimators improve at each episode. Furthermore, the analysis requires redefining the notion of diameter of the POMDP. In addition, we carefully bound the various perturbation terms in order to obtain efficient scaling in terms of dimensionality factors.

Finally, in the Appendix 5.5, we report preliminary synthetic experiments that demonstrate

[2]This principle has been successfully used in a wide number of exploration–exploitation problems ranging from multi-armed bandit (Auer et al., 2002), linear contextual bandit (Abbasi-Yadkori et al., 2011), linear quadratic control (Abbasi-Yadkori and Szepesvári, 2011), and reinforcement learning (Ortner and Auer, 2007; Jaksch et al., 2010b).

superiority of our method over existing RL methods such as Q-learning and UCRL for MDPs, and also over purely exploratory methods such as random sampling, which randomly chooses actions independent of the observations. SM-UCRL converges much faster and to a better solution. The solutions relying on the MDP assumption, directly work in the (high) dimensional observation space and perform poorly. In fact, they can even be worse than the random sampling policy baseline. In contrast, our method aims to find the lower dimensional latent space to derive the policy and this allows UCRL to find a much better memoryless policy with vanishing regret.

5.1.2 Related Work

While RL in MDPs has been widely studied (Kearns and Singh, 2002; Brafman and Tennenholtz, 2003; Bartlett and Tewari, 2009; Jaksch et al., 2010b), the design of effective exploration–exploration strategies in POMDPs is still relatively unexplored. Ross et al. (2007) and Poupart and Vlassis (2008) propose to integrate the problem of estimating the belief state into a model-based Bayesian RL approach, where a distribution over possible MDPs is updated over time. The proposed algorithms are such that the Bayesian inference can be done accurately and at each step, a POMDP is sampled from the posterior and the corresponding optimal policy is executed. While the resulting methods implicitly balance exploration and exploitation, no theoretical guarantee is provided about their regret and their algorithmic complexity requires the introduction of approximation schemes for both the inference and the planning steps. An alternative to model-based approaches is to adapt model-free algorithms, such as Q-learning, to the case of POMDPs. Perkins (2002) proposes a Monte-Carlo approach to action-value estimation and it shows convergence to locally optimal memoryless policies. While this algorithm has the advantage of being computationally efficient, local optimal policies may be arbitrarily suboptimal and thus suffer a linear regret.

An alternative approach to solve POMDPs is to use policy search methods, which avoid estimating value functions and directly optimize the performance by searching in a given policy space, which usually contains memoryless policies (see e.g., (Ng and Jordan, 2000),(Baxter and Bartlett, 2001a),(Poupart and Boutilier, 2003; Bagnell et al., 2004)). Beside its practical success in offline problems, policy search has been successfully integrated with efficient exploration–exploitation techniques and shown to achieve small regret (Gheshlaghi-Azar et al., 2013, 2014). Nonetheless, the performance of such methods is severely constrained by the choice of the policy space, which may not contain policies with good performance.

Matrix decomposition methods have been previously used in the more general setting of predictive state representation (PSRs) (Boots et al., 2011) to reconstruct the structure of the dynamical system. Despite the generality of PSRs, the proposed model relies on strong assumptions on the dynamics of the system and it does not have any theoretical guarantee about its performance. Gheshlaghi azar et al. (2013) used spectral tensor decomposition methods in the multi-armed bandit framework to identify the hidden generative model of a sequence of bandit problems and showed that this may drastically reduce the regret.

Krishnamurthy et al. (2016a) recently analyzed the problem of learning in contextual-MDPs and proved sample complexity bounds polynomial in the capacity of the policy space, the number of states, and the horizon. While their objective is to minimize the regret over a finite horizon, we instead consider the infinite horizon problem. It is an open question to analyze and modify our spectral UCRL algorithm for the finite horizon problem. As stated earlier, contextual MDPs are a special class of POMDPs for which memoryless policies are optimal. While they assume that the samples are drawn from a contextual MDP, we can handle a much more general class of POMDPs, and we minimize regret with respect to the best memoryless policy for the given POMDP.

Finally, a related problem is considered by Ortner et al. (2014), where a series of possible representations based on observation histories is available to the agent but only one of them

is actually Markov. A UCRL-like strategy is adopted and shown to achieve near-optimal regret.

In this paper, we focus on the learning problem, while we consider access to an optimization oracle to compute the optimal memoryless policy. The problem of planning in general POMDPs is intractable (PSPACE-complete for finite horizon (Papadimitriou and Tsitsiklis, 1987a) and uncomputable for infinite horizon (Madani, 1998)).Many exact, approximate, and heuristic methods have been proposed to compute the optimal policy (see Spaan (2012) for a recent survey). An alternative approach is to consider memoryless policies which directly map observations (or a finite history) to actions (Littman, 1994; Singh et al., 1994; Li et al., 2011). While deterministic policies may perform poorly, stochastic memoryless policies are shown to be near-optimal in many domains (Barto et al., 1983; Loch and Singh, 1998; Williams and Singh, 1998) and even optimal in the specific case of contextual MDPs (Krishnamurthy et al., 2016a). Although computing the optimal stochastic memoryless policy is still NP-hard (Littman, 1994), several model-based and model-free methods are shown to converge to nearly-optimal policies with polynomial complexity under some conditions on the POMDP (Jaakkola et al., 1995; Li et al., 2011). In this work, we employ memoryless policies and prove regret bounds for reinforcement learning of POMDPs. The above works suggest that focusing to memoryless policies may not be a restrictive limitation in practice.

5.1.3 Paper Organization

The paper is organized as follows. Sect. 5.2 introduces the notation and the technical assumptions concerning the POMDP and the space of memoryless policies that we consider. Sect. 5.3 introduces the spectral method for the estimation of POMDP parameters together with Thm. 5.1. In Sect. 5.4, we outline SM-UCRL where we integrate the spectral method into an exploration–exploitation strategy and we prove the regret bound of Thm. 5.2. Sect. 5.6

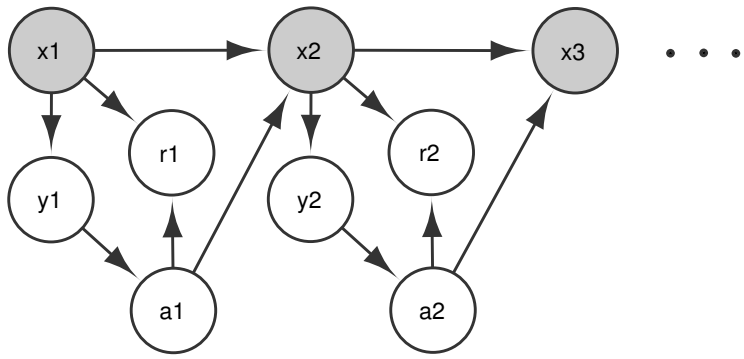


Figure 5.1: Graphical model of a POMDP under memoryless policies.

draws conclusions and discuss possible directions for future investigation. The proofs are reported in the appendix together with preliminary empirical results showing the effectiveness of the proposed method.

5.2 Preliminaries

A POMDP M is a tuple $\langle \mathcal{X}, \mathcal{A}, \mathcal{Y}, \mathcal{R}, f_T, f_R, f_O \rangle$, where $\mathcal{X}$ is a finite state space with cardinality $|\mathcal{X}| = X$, $\mathcal{A}$ is a finite action space with cardinality $|\mathcal{A}| = A$, $\mathcal{Y}$ is a finite observation space with cardinality $|\mathcal{Y}| = Y$, and $\mathcal{R}$ is a finite reward space with cardinality $|\mathcal{R}| = R$ and largest reward $r_{\max}$. For notation convenience, we use a vector notation for the elements in $\mathcal{Y}$ and $\mathcal{R}$, so that $\vec{y} \in \mathbb{R}^Y$ and $\vec{r} \in \mathbb{R}^R$ are indicator vectors with entries equal to 0 except a 1 in the position corresponding to a specific element in the set (e.g., $\vec{y} = \vec{e}_n$ refers to the n-th element in $\mathcal{Y}$). We use $i, j \in [X]$ to index states, $k, l \in [A]$ for actions, $m \in [R]$ for rewards, and $n \in [Y]$ for observations. Finally, f_T denotes the transition density, so that $f_T(x'|x, a)$ is the probability of transition to x' given the state-action pair (x, a), f_R is the reward density, so that $f_R(\vec{r}|x, a)$ is the probability of receiving the reward in $\mathcal{R}$ corresponding to the value of the indicator vector $\vec{r}$ given the state-action pair (x, a), and f_O is the observation density, so that $f_O(\vec{y}|x)$ is the probability of receiving the observation in $\mathcal{Y}$ corresponding to the indicator vector $\vec{y}$ given the state x. Whenever convenient, we use tensor forms for the

density functions such that

$$
\begin{aligned}
T_{i,j,l} &= \mathbb{P}[x_{t+1} = j | x_t = i, a_t = l] = f_T(j|i,l), && s.t.\ \ T \in \mathbb{R}^{X \times X \times A} \\
O_{n,i} &= \mathbb{P}[\vec{y} = \vec{e}_n | x = i] = f_O(\vec{e}_n | i), && s.t.\ \ O \in \mathbb{R}^{Y \times X} \\
\Gamma_{i,l,m} &= \mathbb{P}[\vec{r} = \vec{e}_m | x = i, a = l] = f_R(\vec{e}_m | i, l), && s.t.\ \ \Gamma \in \mathbb{R}^{X \times A \times R}.
\end{aligned}
$$

We also denote by $T_{:,j,l}$ the fiber (vector) in $\mathbb{R}^X$ obtained by fixing the arrival state j and action l and by $T_{:,:,l} \in \mathbb{R}^{X \times X}$ the transition matrix between states when using action l. The graphical model associated to the POMDP is illustrated in Fig. 5.1.

We focus on stochastic memoryless policies which map observations to actions and for any policy π we denote by $f_\pi(a|\vec{y})$ its density function. We denote by $\mathcal{P}$ the set of all stochastic memoryless policies that have a non-zero probability to explore all actions:

$$
\mathcal{P} = \{\pi : \min_{\vec{y}} \min_a f_\pi(a|\vec{y}) > \pi_{\min}\}.
$$

Acting according to a policy π in a POMDP M defines a Markov chain characterized by a transition density

$$
f_{T,\pi}(x'|x) = \sum_a \sum_{\vec{y}} f_\pi(a|\vec{y}) f_O(\vec{y}|x) f_T(x'|x,a),
$$

and a stationary distribution ω_π over states such that $\omega_\pi(x) = \sum_{x'} f_{T,\pi}(x'|x)\omega_\pi(x')$. The expected average reward performance of a policy π is

$$
\eta(\pi; M) = \sum_x \omega_\pi(x) \overline{r}_\pi(x),
$$

where $\overline{r}_\pi(x)$ is the expected reward of executing policy π in state x defined as

$$\overline{r}_\pi(x) = \sum_a \sum_{\vec{y}} f_O(\vec{y}|x) f_\pi(a|\vec{y}) \overline{r}(x,a),$$

and $\overline{r}(x,a) = \sum_r r f_R(r|x,a)$ is the expected reward for the state-action pair (x,a). The best stochastic memoryless policy in $\mathcal{P}$ is $\pi^+ = \arg\max_{\pi \in \mathcal{P}} \eta(\pi; M)$ and we denote by $\eta^+ = \eta(\pi^+; M)$ its average reward.[3] Throughout the paper we assume that we have access to an optimization oracle returning the optimal policy π^+ in $\mathcal{P}$ for any POMDP M. We need the following assumptions on the POMDP M.

Assumption 3 (Ergodicity)**.** *For any policy $\pi \in \mathcal{P}$, the corresponding Markov chain $f_{T,\pi}$ is ergodic, so $\omega_\pi(x) > 0$ for all states $x \in \mathcal{X}$.*

We further characterize the Markov chains that can be generated by the policies in $\mathcal{P}$. For any ergodic Markov chain with stationary distribution ω_π, let $f_{1\to t}(x_t|x_1)$ by the distribution over states reached by a policy π after t steps starting from an initial state x_1. The inverse mixing time $\rho_{\text{mix},\pi}(t)$ of the chain is defined as

$$\rho_{\text{mix},\pi}(t) = \sup_{x_1} \|f_{1\to t}(\cdot|x_1) - \omega_\pi\|_{\text{TV}},$$

where $\|\cdot\|_{\text{TV}}$ is the total-variation metric. Kontorovich et al. (2014) show that for any ergodic Markov chain the mixing time can be bounded as

$$\rho_{\text{mix},\pi}(t) \leq G(\pi)\theta^{t-1}(\pi),$$

where $1 \leq G(\pi) < \infty$ is the *geometric ergodicity* and $0 \leq \theta(\pi) < 1$ is the *contraction coefficient* of the Markov chain generated by policy π.

[3] We use π^+ rather than π^* to recall the fact that we restrict the attention to $\mathcal{P}$ and the actual optimal policy for a POMDP in general should be constructed on the belief-MDP.

Assumption 4 (Full Column-Rank)**.** *The observation matrix $O \in \mathbb{R}^{Y \times X}$ is full column rank.*

and define

This assumption guarantees that the distribution $f_O(\cdot|x)$ in a state x (i.e., a column of the matrix O) is not the result of a linear combination of the distributions over other states. We show later that this is a sufficient condition to recover f_O since it makes all states *distinguishable* from the observations and it also implies that $Y \geq X$. Notice that POMDPs have been often used in the opposite scenario ($X \gg Y$) in applications such as robotics, where imprecise sensors prevents from distinguishing different states. On the other hand, there are many domains in which the number of observations may be much larger than the set of states that define the dynamics of the system. A typical example is the case of spoken dialogue systems (Atrash and Pineau, 2006; Png et al., 2012), where the observations (e.g., sequences of words uttered by the user) is much larger than the state of the conversation (e.g., the actual meaning that the user intended to communicate). A similar scenario is found in medical applications (Hauskrecht and Fraser, 2000), where the state of a patient (e.g., sick or healthy) can produce a huge body of different (random) observations. In these problems it is crucial to be able to reconstruct the underlying small state space and the actual dynamics of the system from the observations.

Assumption 5 (Invertible)**.** *For any action $a \in [A]$, the transition matrix $T_{:,:,a} \in \mathbb{R}^{X \times X}$ is invertible.*

Similar to the previous assumption, this means that for any action a the distribution $f_T(\cdot|x, a)$ cannot be obtained as linear combination of distributions over other states, and it is a sufficient condition to be able to recover the transition tensor. Both Asm. 4 and 5 are strictly related to the assumptions introduced by Anandkumar et al. (2014) for tensor methods in HMMs. In Sect. 5.4 we discuss how they can be partially relaxed.

5.3 Learning the Parameters of the POMDP

In this section we introduce a novel spectral method to estimate the POMDP parameters f_T, f_O, and f_R. A stochastic policy π is used to generate a trajectory $(\vec{y}_1, a_1, \vec{r}_1, \ldots, \vec{y}_N, a_N, \vec{r}_N)$ of N steps. We need the following assumption that, together with Asm. 3, guarantees that all states and actions are constantly visited.

Assumption 6 (Policy Set)**.** *The policy π belongs to $\mathcal{P}$.*

Similar to the case of HMMs, the key element to apply the spectral methods is to construct a multi-view model for the hidden states. Despite its similarity, the spectral method developed for HMM by Anandkumar et al. (2014) cannot be directly employed here. In fact, in HMMs the state transition and the observations only depend on the current state. On the other hand, in POMDPs the probability of a transition to state x' not only depends on x, but also on action a. Since the action is chosen according to a memoryless policy π based on the current observation, this creates an indirect dependency of x' on observation $\vec{y}$, which makes the model more intricate.

5.3.1 The multi-view model

We estimate POMDP parameters for each action $l \in [A]$ separately. Let $t \in [2, N-1]$ be a step at which $a_t = l$, we construct three views $(a_{t-1}, \vec{y}_{t-1}, \vec{r}_{t-1})$, $(\vec{y}_t, \vec{r}_t)$, and $(\vec{y}_{t+1})$ which all contain observable elements. As it can be seen in Fig. 5.1, all three views provide some information about the hidden state x_t (e.g., the observation $\vec{y}_{t-1}$ triggers the action a_{t-1}, which influence the transition to x_t). A careful analysis of the graph of dependencies shows that conditionally on x_t, a_t all the views are independent. For instance, let us consider $\vec{y}_t$ and $\vec{y}_{t+1}$. These two random variables are clearly dependent since $\vec{y}_t$ influences action a_t, which triggers a transition to x_{t+1} that emits an observation $\vec{y}_{t+1}$. Nonetheless, it is sufficient to

condition on the action $a_t = l$ to break the dependency and make $\vec{y}_t$ and $\vec{y}_{t+1}$ independent. Similar arguments hold for all the other elements in the views, which can be used to recover the latent variable x_t. More formally, we encode the triple $(a_{t-1}, \vec{y}_{t-1}, \vec{r}_{t-1})$ into a vector $\vec{v}_{1,t}^{(l)} \in \mathbb{R}^{A \cdot Y \cdot R}$, so that view $\vec{v}_{1,t}^{(l)} = \vec{e}_s$ whenever $a_{t-1} = k$, $\vec{y}_{t-1} = \vec{e}_n$, and $\vec{r}_{t-1} = \vec{e}_m$ for a suitable mapping between the index $s \in \{1, \dots, A \cdot Y \cdot R\}$ and the indices (k, n, m) of the action, observation, and reward. Similarly, we proceed for $\vec{v}_{2,t}^{(l)} \in \mathbb{R}^{Y \cdot R}$ and $\vec{v}_{3,t}^{(l)} \in \mathbb{R}^{Y}$. We introduce the three view matrices $V_\nu^{(l)}$ with $\nu \in \{1, 2, 3\}$ associated with action l defined as $V_1^{(l)} \in \mathbb{R}^{A \cdot Y \cdot R \times X}$, $V_2^{(l)} \in \mathbb{R}^{Y \cdot R \times X}$, and $V_3^{(l)} \in \mathbb{R}^{Y \times X}$ such that

$$
\begin{aligned}
[V_1^{(l)}]_{s,i} &= \mathbb{P}\big(\vec{v}_1^{(l)} = \vec{e}_s | x_2 = i\big) = [V_1^{(l)}]_{(n,m,k),i} = \mathbb{P}\big(\vec{y}_1 = \vec{e}_n, \vec{r}_1 = \vec{e}_m, a_1 = k | x_2 = i\big), \\
[V_2^{(l)}]_{s,i} &= \mathbb{P}\big(\vec{v}_2^{(l)} = \vec{e}_s | x_2 = i, a_2 = l\big) = [V_2^{(l)}]_{(n',m'),i} = \mathbb{P}\big(\vec{y}_2 = \vec{e}_{n'}, \vec{r}_2 = \vec{e}_{m'} | x_2 = i, a_2 = l\big), \\
[V_3^{(l)}]_{s,i} &= \mathbb{P}\big(\vec{v}_3^{(l)} = \vec{e}_s | x_2 = i, a_2 = l\big) = [V_3^{(l)}]_{n'',i} = \mathbb{P}\big(\vec{y}_3 = \vec{e}_{n''} | x_2 = i, a_2 = l\big).
\end{aligned}
$$

In the following we denote by $\mu_{\nu,i}^{(l)} = [V_\nu^{(l)}]_{:,i}$ the ith column of the matrix $V_\nu^{(l)}$ for any $\nu \in \{1, 2, 3\}$. Notice that Asm. 4 and Asm. 5 imply that all the view matrices are full column rank. As a result, we can construct a multi-view model that relates the spectral decomposition of the second and third moments of the (modified) views with the columns of the third view matrix.

Proposition 1 (Thm. 3.6 in (Anandkumar et al., 2014))**.** *Let $K_{\nu,\nu'}^{(l)} = \mathbb{E}\big[\vec{v}_\nu^{(l)} \otimes \vec{v}_{\nu'}^{(l)}\big]$ be the correlation matrix between views ν and ν' and $K^\dagger$ is its pseudo-inverse. We define a modified version of the first and second views as*

$$
\widetilde{\vec{v}}_1^{(l)} := K_{3,2}^{(l)} (K_{1,2}^{(l)})^\dagger \vec{v}_1^{(l)}, \quad \widetilde{\vec{v}}_2^{(l)} := K_{3,1}^{(l)} (K_{2,1}^{(l)})^\dagger \vec{v}_2^{(l)}. \tag{5.1}
$$

Then the second and third moment of the modified views have a spectral decomposition as

$$M_2^{(l)} = \mathbb{E}\big[\widetilde{\vec{v}}_1^{(l)} \otimes \widetilde{\vec{v}}_2^{(l)}\big] = \sum_{i=1}^{X} \omega_\pi^{(l)}(i)\mu_{3,i}^{(l)} \otimes \mu_{3,i}^{(l)}, \tag{5.2}$$

$$M_3^{(l)} = \mathbb{E}\big[\widetilde{\vec{v}}_1^{(l)} \otimes \widetilde{\vec{v}}_2^{(l)} \otimes \vec{v}_3^{(l)}\big] = \sum_{i=1}^{X} \omega_\pi^{(l)}(i)\mu_{3,i}^{(l)} \otimes \mu_{3,i}^{(l)} \otimes \mu_{3,i}^{(l)}, \tag{5.3}$$

where $\otimes$ is the tensor product and $\omega_\pi^{(l)}(i) = \mathbb{P}[x = i|a = l]$ is the state stationary distribution of π conditioned on action l being selected by policy π.

Notice that under Asm. 3 and 6, $\omega_\pi^{(l)}(i)$ is always bounded away from zero. Given $M_2^{(l)}$ and $M_3^{(l)}$ we can recover the columns of the third view $\mu_{3,i}^{(l)}$ directly applying the standard spectral decomposition method of Anandkumar et al. (2012). We need to recover the other views from $V_3^{(l)}$. From the definition of modified views in Eq. 5.1 we have

$$\begin{aligned} \mu_{3,i}^{(l)} &= \mathbb{E}\big[\widetilde{\vec{v}}_1|x_2 = i, a_2 = l\big] = K_{3,2}^{(l)}(K_{1,2}^{(l)})^\dagger \mathbb{E}\big[\vec{v}_1|x_2 = i, a_2 = l\big] = K_{3,2}^{(l)}(K_{1,2}^{(l)})^\dagger \mu_{1,i}^{(l)}, \\ \mu_{3,i}^{(l)} &= \mathbb{E}\big[\widetilde{\vec{v}}_2|x_2 = i, a_2 = l\big] = K_{3,1}^{(l)}(K_{2,1}^{(l)})^\dagger \mathbb{E}\big[\vec{v}_2|x_2 = i, a_2 = l\big] = K_{3,1}^{(l)}(K_{2,1}^{(l)})^\dagger \mu_{2,i}^{(l)}. \end{aligned} \tag{5.4}$$

Thus, it is sufficient to invert (pseudo invert) the two equations above to obtain the columns of both the first and second view matrices. This process could be done in any order, e.g., we could first estimate the second view by applying a suitable symmetrization step (Eq. 5.1) and recovering the first and the third views by reversing similar equations to Eq. 5.4. On the other hand, we cannot repeat the symmetrization step multiple times and estimate the views independently (i.e., without inverting Eq. 5.4). In fact, the estimates returned by the spectral method are consistent "up to a suitable permutation" on the indexes of the states. While this does not pose any problem in computing one single view, if we estimated two views independently, the permutation may be different, thus making them non-consistent and impossible to use in recovering the POMDP parameters. On the other hand, estimating first one view and recovering the others by inverting Eq. 5.4 guarantees the consistency of the labeling of the hidden states.

5.3.2 Recovery of POMDP parameters

Once the views $\{V_\nu^{(l)}\}_{\nu=2}^3$ are computed from $M_2^{(l)}$ and $M_3^{(l)}$, we can derive f_T, f_O, and f_R. In particular, all parameters of the POMDP can be obtained by manipulating the second and third view as illustrated in the following lemma.

Lemma 2. *Given the views $V_2^{(l)}$ and $V_3^{(l)}$, for any state $i \in [X]$ and action $l \in [A]$, the POMDP parameters are obtained as follows. For any reward $m \in [R]$ the reward density is*

$$f_R(\vec{e}_{m'}|i,l) = \sum_{n'=1}^{Y} [V_2^{(l)}]_{(n',m'),i}; \tag{5.5}$$

for any observation $n' \in [Y]$ the observation density is

$$f_O^{(l)}(\vec{e}_{n'}|i) = \sum_{m'=1}^{R} \frac{[V_2^{(l)}]_{(n',m'),i}}{f_\pi(l|\vec{e}_{n'})\rho(i,l)}, \tag{5.6}$$

with

$$\rho(i,l) = \sum_{m'=1}^{R}\sum_{n'=1}^{Y} \frac{[V_2^{(l)}]_{(n',m'),i}}{f_\pi(l|\vec{e}_{n'})} = \frac{1}{\mathbb{P}(a_2 = l|x_2 = i)}.$$

Finally, each second mode of the transition tensor $T \in \mathbb{R}^{X\times X\times A}$ is obtained as

$$[T]_{i,:,l} = O^\dagger [V_3^{(l)}]_{:,i}, \tag{5.7}$$

where $O^\dagger$ is the pseudo-inverse of matrix observation O and $f_T(\cdot|i,l) = [T]_{i,:,l}$.

In the previous statement we use $f_O^{(l)}$ to denote that the observation model is recovered from the view related to action l. While in the exact case, all $f_O^{(l)}$ are identical, moving to the empirical version leads to A different estimates, one for each action view used to compute it. Among them, we will select the estimate with the better accuracy.

[4]Each column of $O^{(l)}$ corresponds to $\ell 1$-closest column of $O^{(l^*)}$

Empirical estimates of POMDP parameters. In practice, $M_2^{(l)}$ and $M_3^{(l)}$ are not available and need to be estimated from samples. Given a trajectory of N steps obtained executing policy π, let $\mathcal{T}(l) = \{t \in [2, N-1] : a_t = l\}$ be the set of steps when action l is played, then we collect all the triples $(a_{t-1}, \vec{y}_{t-1}, \vec{r}_{t-1})$, $(\vec{y}_t, \vec{r}_t)$ and $(\vec{y}_{t+1})$ for any $t \in \mathcal{T}(l)$ and construct the corresponding views $\vec{v}_{1,t}^{(l)}$, $\vec{v}_{2,t}^{(l)}$, $\vec{v}_{3,t}^{(l)}$. Then we symmetrize the views using empirical estimates of the covariance matrices and build the empirical version of Eqs. 5.2 and 5.3 using $N(l) = |\mathcal{T}(l)|$ samples, thus obtaining

$$\widehat{M}_2^{(l)} = \frac{1}{N(l)} \sum_{t \in \mathcal{T}_l} \tilde{\vec{v}}_{1,t}^{(l)} \otimes \tilde{\vec{v}}_{2,t}^{(l)}, \qquad \widehat{M}_3^{(l)} = \frac{1}{N(l)} \sum_{t \in \mathcal{T}_l} \tilde{\vec{v}}_{1,t}^{(l)} \otimes \tilde{\vec{v}}_{2,t}^{(l)} \otimes \vec{v}_{3,t}^{(l)}. \tag{5.8}$$

Given the resulting $\widehat{M}_2^{(l)}$ and $\widehat{M}_3^{(l)}$, we apply the spectral tensor decomposition method to recover an empirical estimate of the third view $\widehat{V}_3^{(l)}$ and invert Eq. 5.4 (using estimated covariance matrices) to obtain $\widehat{V}_2^{(l)}$. Finally, the estimates $\widehat{f}_O$, $\widehat{f}_T$, and $\widehat{f}_R$ are obtained by plugging the estimated views $\widehat{V}_\nu$ in the process described in Lemma 2.

Spectral methods indeed recover the factor matrices up to a permutation of the hidden states. In this case, since we separately carry out spectral decompositions for different actions, we recover permuted factor matrices. Since the observation matrix O is common to all the actions, we use it to align these decompositions. Let's define d_O

$$d_O =: \min_{x,x'} \|f_O(\cdot|x) - f_O(\cdot|x')\|_1$$

Actually, d_O is the minimum separability level of matrix O. When the estimation error over columns of matrix O are less than $4d_O$, then one can come over the permutation issue by matching columns of O^l matrices. In T condition is reflected as a condition that the number of samples for each action has to be larger some number.

The overall method is summarized in Alg. 6. The empirical estimates of the POMDP parameters enjoy the following guarantee.

Theorem 5.1 (Learning Parameters). *Let $\widehat{f}_O$, $\widehat{f}_T$, and $\widehat{f}_R$ be the estimated POMDP models using a trajectory of N steps. We denote by $\sigma^{(l)}_{\nu,\nu'} = \sigma_X(K^{(l)}_{\nu,\nu'})$ the smallest non-zero singular value of the covariance matrix $K_{\nu,\nu'}$, with $\nu, \nu' \in \{1,2,3\}$, and by $\sigma_{\min}(V^{(l)}_\nu)$ the smallest singular value of the view matrix $V^{(l)}_\nu$ (strictly positive under Asm. 4 and Asm. 5), and we define $\omega^{(l)}_{\min} = \min_{x\in\mathcal{X}} \omega^{(l)}_\pi(x)$ (strictly positive under Asm. 3). If for any action $l \in [A]$, the number of samples $N(l)$ satisfies the condition*

$$N(l) \geq \max\left\{\frac{4}{(\sigma^{(l)}_{3,1})^2}, \frac{16C_O^2 YR}{{\lambda^{(l)}}^2 d_O^2}, \left(\frac{G(\pi)\frac{2\sqrt{2}+1}{1-\theta(\pi)}}{\omega^{(l)}_{\min} \min\limits_{\nu\in\{1,2,3\}}\{\sigma^2_{\min}(V^{(l)}_\nu)\}}\right)^2 \Theta^{(l)}\right\} \log\left(\frac{2(Y^2+AYR)}{\delta}\right), \tag{5.9}$$

with $\Theta^{(l)}$, defined in Eq 5.9[5]*, and $G(\pi), \theta(\pi)$ are the geometric ergodicity and the contraction coefficients of the corresponding Markov chain induced by π, then for any $\delta \in (0,1)$ and for any state $i \in [X]$ and action $l \in [A]$ we have*

$$\|\widehat{f}^{(l)}_O(\cdot|i) - f_O(\cdot|i)\|_1 \leq \mathcal{B}^{(l)}_O := \frac{C_O}{\lambda^{(l)}}\sqrt{\frac{YR\log(1/\delta)}{N(l)}}, \tag{5.10}$$

$$\|\widehat{f}_R(\cdot|i,l) - f_R(\cdot|i,l)\|_1 \leq \mathcal{B}^{(l)}_R := \frac{C_R}{\lambda^{(l)}}\sqrt{\frac{YR\log(1/\delta)}{N(l)}}, \tag{5.11}$$

$$\|\widehat{f}_T(\cdot|i,l) - f_T(\cdot|i,l)\|_2 \leq \mathcal{B}^{(l)}_T := \frac{C_T}{\lambda^{(l)}}\sqrt{\frac{YRX^2\log(1/\delta)}{N(l)}}, \tag{5.12}$$

with probability $1-6(Y^2+AYR)A\delta$ (w.r.t. the randomness in the transitions, observations,

[5] We do not report the explicit definition of $\Theta^{(l)}$ here because it contains exactly the same quantities, such as $\omega^{(l)}_{\min}$, that are already present in other parts of the condition of Eq. 5.9.

and policy), where C_O, C_R, and C_T are numerical constants and

$$\lambda^{(l)} = \sigma_{\min}(O)(\pi^{(l)}_{\min})^2 \sigma^{(l)}_{1,3}(\omega^{(l)}_{\min} \min_{\nu \in \{1,2,3\}} \{\sigma^2_{\min}(V^{(l)}_{\nu})\})^{3/2}. \tag{5.13}$$

Finally, we denote by $\widehat{f}_O$ the most accurate estimate of the observation model, i.e., the estimate $\widehat{f}^{(l^*)}_O$ such that $l^* = \arg\min_{l \in [A]} \mathcal{B}^{(l)}_O$ and we denote by $\mathcal{B}_O$ its corresponding bound.

Remark 1 (consistency and dimensionality). All previous errors decrease with a rate $\widetilde{O}(1/\sqrt{N(l)})$, showing the consistency of the spectral method, so that if all the actions are repeatedly tried over time, the estimates converge to the true parameters of the POMDP. This is in contrast with EM-based methods which typically get stuck in local maxima and return biased estimators, thus preventing from deriving confidence intervals.

The bounds in Eqs. 5.10, 5.11, 5.12 on $\widehat{f}_O$, $\widehat{f}_R$ and $\widehat{f}_T$ depend on X, Y, and R (and the number of actions only appear in the probability statement). The bound in Eq. 5.12 on $\widehat{f}_T$ is worse than the bounds for $\widehat{f}_R$ and $\widehat{f}_O$ in Eqs. 5.10, 5.11 by a factor of X^2. This seems unavoidable since $\widehat{f}_R$ and $\widehat{f}_O$ are the results of the manipulation of the matrix $V^{(l)}_2$ with $Y \cdot R$ columns, while estimating $\widehat{f}_T$ requires working on both $V^{(l)}_2$ and $V^{(l)}_3$. In addition, to come up with upper bound for $\widehat{f}_T$, more complicated bound derivation is needed and it has one step of Frobenious norms to $\ell 2$ norm transformation. The derivation procedure for $\widehat{f}_T$ is more complicated compared to $\widehat{f}_O$ and $\widehat{f}_R$ and adds the term X to the final bound. (Appendix in Azizzadenesheli et al. (2016c))

Remark 2 (POMDP parameters and policy π). In the previous bounds, several terms depend on the structure of the POMDP and the policy π used to collect the samples:

- $\lambda^{(l)}$ captures the main problem-dependent terms. While $K_{1,2}$ and $K_{1,3}$ are full column-rank matrices (by Asm. 4 and 5), their smallest non-zero singular values influence the

accuracy of the (pseudo-)inversion in the construction of the modified views in Eq. 5.1 and in the computation of the second view from the third using Eq. 5.4. Similarly the presence of $\sigma_{\min}(O)$ is justified by the pseudo-inversion of O used to recover the transition tensor in Eq. 5.7. Finally, the dependency on the smallest singular values $\sigma^2_{\min}(V_\nu^{(l)})$ is due to the tensor decomposition method (see the Appendix of Azizzadenesheli et al. (2016c)).

- A specific feature of the bounds above is that they do not depend on the state i and the number of times it has been explored. Indeed, the inverse dependency on $\omega^{(l)}_{\min}$ in the condition on $N(l)$ in Eq. 5.9 implies that if a state j is poorly visited, then the empirical estimate of any other state i may be negatively affected. This is in striking contrast with the fully observable case where the accuracy in estimating, e.g., the reward model in state i and action l, simply depends on the number of times that state-action pair has been explored, even if some other states are never explored at all. This difference is intrinsic in the partial observable nature of the POMDP, where we reconstruct information about the states (i.e., reward, transition, and observation models) only from indirect observations. As a result, in order to have accurate estimates of the POMDP structure, we need to rely on the policy π and the ergodicity of the corresponding Markov chain to guarantee that the whole state space is covered.

- Under Asm. 3 the Markov chain $f_{T,\pi}$ is ergodic for any $\pi \in \mathcal{P}$. Since no assumption is made on the fact that the samples generated from π being sampled from the stationary distribution, the condition on $N(l)$ depends on how fast the chain converge to ω_π and this is characterized by the parameters $G(\pi)$ and $\theta(\pi)$.

- If the policy is deterministic, then some actions would not be explored at all, thus leading to very inaccurate estimations (see e.g., the dependency on $f_\pi(l|\vec{y})$ in Eq. 5.6). The inverse dependency on $\pi_{\min}$ (defined in $\mathcal{P}$) accounts for the amount of exploration assigned to every actions, which determines the accuracy of the estimates. Further-

more, notice that also the singular values $\sigma_{1,3}^{(l)}$ and $\sigma_{1,2}^{(l)}$ depend on the distribution of the views, which in turn is partially determined by the policy π.

Notice that the first two terms are basically the same as in the bounds for spectral methods applied to HMM (Song et al., 2013), while the dependency on $\pi_{\min}$ is specific to the POMDP case. On the other hand, in the analysis of HMMs usually there is no dependency on the parameters G and θ because the samples are assumed to be drawn from the stationary distribution of the chain. Removing this assumption required developing novel results for the tensor decomposition process itself using extensions of matrix concentration inequalities for the case of Markov chain (not yet in the stationary distribution). The overall analysis is reported in the Appendix of Azizzadenesheli et al. (2016c). It worth to note that, Kontorovich et al. (2013), without stationary assumption, proposes new method to learn the transition matrix of HMM model given factor matrix O, and it provides theoretical bound over estimation errors.

5.4 Spectral UCRL

The most interesting aspect of the estimation process illustrated in the previous section is that it can be applied when samples are collected using any policy π in the set $\mathcal{P}$. As a result, it can be integrated into any exploration-exploitation strategy where the policy changes over time in the attempt of minimizing the regret.

The algorithm. The SM-UCRL algorithm illustrated in Alg. 7 is the result of the integration of the spectral method into a structure similar to UCRL (Jaksch et al., 2010b) designed to optimize the exploration-exploitation trade-off. The learning process is split into episodes of increasing length. At the beginning of each episode $k > 1$ (the first episode is used to initialize the variables), an estimated POMDP $\widehat{M}^{(k)} = (X, A, Y, R, \widehat{f}_T^{(k)}, \widehat{f}_R^{(k)}, \widehat{f}_O^{(k)})$ is

computed using the spectral method of Alg. 6. Unlike in UCRL, SM-UCRL cannot use all the samples from past episodes. In fact, the distribution of the views $\vec{v}_1, \vec{v}_2, \vec{v}_3$ depends on the policy used to generate the samples. As a result, whenever the policy changes, the spectral method should be re-run using only the samples collected by that specific policy. Nonetheless we can exploit the fact that the spectral method is applied to each action separately. In SM-UCRL at episode k for each action l we use the samples coming from the past episode which returned the largest number of samples for that action. Let $v^{(k)}(l)$ be the number of samples obtained during episode k for action l, we denote by $N^{(k)}(l) = \max_{k'<k} v^{(k')}(l)$ the largest number of samples available from past episodes for each action separately and we feed them to the spectral method to compute the estimated POMDP $\widehat{M}^{(k)}$ at the beginning of each episode k.

Given the estimated POMDP $\widehat{M}^{(k)}$ and the result of Thm. 5.1, we construct the set $\mathcal{M}^{(k)}$ of *admissible* POMDPs $\widetilde{M} = \langle \mathcal{X}, \mathcal{A}, \mathcal{Y}, \mathcal{R}, \widetilde{f}_T, \widetilde{f}_R, \widetilde{f}_O \rangle$ whose transition, reward, and observation models belong to the confidence intervals (e.g., $\|\widehat{f}_O^{(k)}(\cdot|i) - \widetilde{f}_O(\cdot|i)\|_1 \leq \mathcal{B}_O$ for any state i). By construction, this guarantees that the true POMDP M is included in $\mathcal{M}^{(k)}$ with high probability. Following the *optimism in face of uncertainty* principle used in UCRL, we compute the optimal memoryless policy corresponding to the most optimistic POMDP within $\mathcal{M}^{(k)}$. More formally, we compute[6]

$$\widetilde{\pi}^{(k)} = \arg\max_{\pi \in \mathcal{P}} \max_{M \in \mathcal{M}^{(k)}} \eta(\pi; M). \tag{5.14}$$

Intuitively speaking, the optimistic policy implicitly balances exploration and exploitation. Large confidence intervals suggest that $\widehat{M}^{(k)}$ is poorly estimated and further exploration

[6]The computation of the optimal policy (within $\mathcal{P}$) in the optimistic model may not be trivial. Nonetheless, we first notice that given an horizon N, the policy needs to be recomputed at most $O(\log N)$ times (i.e., number of episodes). Furthermore, if an optimization oracle to $\eta(\pi; M)$ for a given POMDP M is available, then it is sufficient to randomly sample multiple POMDPs from $\mathcal{M}^{(k)}$ (which is a computationally cheap operation), find their corresponding best policy, and return the best among them. If *enough* POMDPs are sampled, the additional regret caused by this approximately optimistic procedure can be bounded as $\widetilde{O}(\sqrt{N})$.

is needed. Instead of performing a purely explorative policy, SM-UCRL still exploits the current estimates to construct the set of admissible POMDPs and selects the policy that maximizes the performance $\eta(\pi; M)$ over all POMDPs in $\mathcal{M}^{(k)}$. The choice of using the optimistic POMDP guarantees the $\widetilde{\pi}^{(k)}$ explores more often actions corresponding to large confidence intervals, thus contributing the improve the estimates over time. After computing the optimistic policy, $\widetilde{\pi}^{(k)}$ is executed until the number of samples for one action is doubled, i.e., $v^{(k)}(l) \geq 2N^{(k)}(l)$. This stopping criterion avoids switching policies too often and it guarantees that when an episode is terminated, enough samples are collected to compute a new (better) policy. This process is then repeated over episodes and we expect the optimistic policy to get progressively closer to the best policy $\pi^+ \in \mathcal{P}$ as the estimates of the POMDP get more and more accurate.

Regret analysis. We now study the regret SM-UCRL w.r.t. the best policy in $\mathcal{P}$. While in general π^+ may not be optimal, $\pi_{\min}$ is usually set to a small value and oftentimes the optimal memoryless policy itself is stochastic and it may actually be contained in $\mathcal{P}$. Given an horizon of N steps, the regret is defined as

$$\mathrm{Reg}_N = N\eta^+ - \sum_{t=1}^{N} r_t, \tag{5.15}$$

where r_t is the random reward obtained at time t according to the reward model f_R over the states traversed by the policies performed over episodes on the actual POMDP. To restate, similar to the MDP case, the complexity of learning in a POMDP M is partially determined by its diameter, defined as

$$D := \max_{x,x' \in \mathcal{X}, a,a' \in \mathcal{A}} \min_{\pi \in \mathcal{P}} \mathbb{E}\big[\tau(x', a' | x, a; \pi)\big], \tag{5.16}$$

which corresponds to the expected passing time from a state x to a state x' starting with action a and terminating with action a' and following the most effective memoryless policy

$\pi \in \mathcal{P}$. The main difference w.r.t. to the diameter of the underlying MDP (see e.g., Jaksch et al. (2010b)) is that it considers the distance between state-action pairs using memoryless policies instead of state-based policies.

Before stating our main result, we introduce the worst-case version of the parameters characterizing Thm. 5.1. Let $\overline{\sigma}_{1,2,3} := \min_{l\in[A]} \min_{\pi\in\mathcal{P}} \omega^{(l)}_{\min} \min_{\nu\in\{1,2,3\}} \sigma^2_{\min}(V^{(l)}_\nu)$ be the worst smallest non-zero singular value of the views for action l when acting according to policy π and let $\overline{\sigma}_{1,3} := \min_{l\in[A]} \min_{\pi\in\mathcal{P}} \sigma_{\min}(K^{(l)}_{1,3}(\pi))$ be the worst smallest non-zero singular value of the covariance matrix $K^{(l)}_{1,3}(\pi)$ between the first and third view for action l when acting according to policy π. Similarly, we define $\overline{\sigma}_{1,2}$. We also introduce $\overline{\omega}_{\min} := \min_{l\in[A]} \min_{x\in[X]} \min_{\pi\in\mathcal{P}} \omega^{(l)}_\pi(x)$ and

$$\overline{N} := \max_{l\in[A]} \max_{\pi\in\mathcal{P}} \max \left\{ \frac{4}{(\overline{\sigma}^2_{3,1})}, \frac{16C_O^2 YR}{{\lambda^{(l)}}^2 d_O^2}, \left(\frac{G(\pi)\frac{2\sqrt{2}+1}{1-\theta(\pi)}}{\overline{\omega}_{\min}\overline{\sigma}_{1,2,3}} \right)^2 \overline{\Theta}^{(l)} \right\} \log\left(2\frac{(Y^2+AYR)}{\delta}\right), \tag{5.17}$$

which is a sufficient number of samples for the statement of Thm. 5.1 to hold for any action and any policy. Here $\overline{\Theta}^{(l)}$ is also model related parameter depending on the underlying model. Then we can prove the following result.

Theorem 5.2 (Regret Bound)**.** *Consider a POMDP M with X states, A actions, Y observations, R rewards, characterized by a diameter D and with an observation matrix $O \in \mathbb{R}^{Y\times X}$ with smallest non-zero singular value $\sigma_X(O)$. We consider the policy space $\mathcal{P}$, such that the worst smallest non-zero value is $\overline{\sigma}_{1,2,3}$ (resp. $\overline{\sigma}_{1,3}$) and the worst smallest probability to reach a state is $\overline{\omega}_{\min}$. If* SM-UCRL *is run over N steps and the confidence intervals of Thm. 5.1 are used with $\delta = \delta'/N^6$ in constructing the plausible POMDPs $\widetilde{\mathcal{M}}$, then under Asm. 3, 4, and 5 it suffers from a total regret*

$$Reg_N \le C_1 \frac{r_{\max}}{\overline{\lambda}} DX^{3/2} \sqrt{AYRN\log(N/\delta')} \tag{5.18}$$

with probability $1-\delta'$, where C_1 is numerical constants, and $\overline{\lambda}$ is the worst-case equivalent

of Eq. 5.13 defined as

$$\overline{\lambda} = \sigma_{\min}(O)\pi_{\min}^2 \overline{\sigma}_{1,3} \overline{\sigma}_{1,2,3}^{3/2}. \tag{5.19}$$

Remark 1 (comparison with MDPs). If UCRL could be run directly on the underlying MDP (i.e., as if the states where directly observable), then it would obtain a regret (Jaksch et al., 2010b)

$$\mathrm{Reg}_N \leq C_{\mathrm{MDP}} D_{\mathrm{MDP}} X \sqrt{AN \log N},$$

where

$$D_{\mathrm{MDP}} := \max_{x,x' \in \mathcal{X}} \min_{\pi} \mathbb{E}[\tau(x'|x;\pi)],$$

with high probability. We first notice that the regret is of order $\widetilde{O}(\sqrt{N})$ in both MDP and POMDP bounds. This means that despite the complexity of POMDPs, SM-UCRL has the same dependency on the number of steps as in MDPs and it has a vanishing per-step regret. Furthermore, this dependency is known to be minimax optimal. The diameter D in general is larger than its MDP counterpart D_{MDP}, since it takes into account the fact that a memoryless policy, that can only work on observations, cannot be as efficient as a state-based policy in moving from one state to another. Although no lower bound is available for learning in POMDPs, we believe that this dependency is unavoidable since it is strictly related to the partial observable nature of POMDPs.

Remark 2 (dependency on POMDP parameters). The dependency on the number of actions is the same in both MDPs and POMDPs. On the other hand, moving to POMDPs naturally brings the dimensionality of the observation and reward models (Y,X, and R

respectively) into the bound. The dependency on Y and R is directly inherited from the bounds in Thm. 5.1. The term $X^{3/2}$ is indeed the results of two terms; X and $X^{1/2}$. The first term is the same as in MDPs, while the second comes from the fact that the transition tensor is derived from Eq. 5.7. Finally, the term $\overline{\lambda}$ in Eq. 5.18 summarizes a series of terms which depend on both the policy space $\mathcal{P}$ and the POMDP structure. These terms are directly inherited from the spectral decomposition method used at the core of SM-UCRL and, as discussed in Sect. 5.3, they are due to the partial observability of the states and the fact that all (unobservable) states need to be visited often enough to be able to compute accurate estimate of the observation, reward, and transition models.

Remark 3 (computability of the confidence intervals). While it is a common assumption that the dimensionality X of the hidden state space is known as well as the number of actions, observations, and rewards, it is not often the case that the terms $\lambda^{(l)}$ appearing in Thm. 5.1 are actually available. While this does not pose any problem for a *descriptive* bound as in Thm. 5.1, in SM-UCRL we actually need to compute the bounds $\mathcal{B}_O^{(l)}$, $\mathcal{B}_R^{(l)}$, and $\mathcal{B}_T^{(l)}$ to explicitly construct confidence intervals. This situation is relatively common in many exploration–exploitation algorithms that require computing confidence intervals containing the range of the random variables or the parameters of their distributions in case of sub-Gaussian variables. In practice these values are often replaced by parameters that are tuned by hand and set to much smaller values than their theoretical ones. As a result, we can run SM-UCRL with the terms $\lambda^{(l)}$ replaced by a fixed parameter. Notice that any inaccurate choice in setting $\lambda^{(l)}$ would mostly translate into bigger multiplicative constants in the final regret bound or in similar bounds but with smaller probability.

In general, computing confidence bound is a hard problem, even for simpler cases such as Markov chains Hsu et al. (2015). Therefore finding upper confidence bounds for POMDP is challenging if we do not know its mixing properties. As it mentioned, another parameter is needed to compute upper confidence bound is $\lambda^{(l)}$ 5.13. As it is described in, in practice,

one can replace the coefficient $\lambda^{(l)}$ with some constant which causes bigger multiplicative constant in final regret bound. Alternatively, one can estimate $\lambda^{(l)}$ from data. In this case, we add a lower order term to the regret which decays as $\frac{1}{N}$.

Remark 4 (relaxation on assumptions). Both Thm. 5.1 and 5.2 rely on the observation matrix $O \in \mathbb{R}^{Y \times X}$ being full column rank (Asm. 4). As discussed in Sect. 5.2 may not be verified in some POMDPs where the number of states is larger than the number of observations ($X > Y$). Nonetheless, it is possible to correctly estimate the POMDP parameters when O is not full column-rank by exploiting the additional information coming from the reward and action taken at step $t+1$. In particular, we can use the triple $(a_{t+1}, \vec{y}_{t+1}, r_{t+1})$ and redefine the third view $V_3^{(l)} \in \mathbb{R}^{d \times X}$ as

$$
\begin{aligned}
[V_3^{(l)}]_{s,i} &= \mathbb{P}(\vec{v}_3^{(l)} = \vec{e}_s | x_2 = i, a_2 = l) = [V_3^{(l)}]_{(n,m,k),i} \\
&= \mathbb{P}(\vec{y}_3 = \vec{e}_n, \vec{r}_3 = \vec{e}_m, a_3 = k | x_2 = i, a_2 = l),
\end{aligned}
$$

and replace Asm. 4 with the assumption that the view matrix $V_3^{(l)}$ is full column-rank, which basically requires having rewards that jointly with the observations are informative enough to reconstruct the hidden state. While this change does not affect the way the observation and the reward models are recovered in Lemma 2, (they only depend on the second view $V_2^{(l)}$), for the reconstruction of the transition tensor, we need to write the third view $V_3^{(l)}$ as

$$
\begin{aligned}
&[V_3^{(l)}]_{s,i} = [V_3^{(l)}]_{(n,m,k),i} \\
&= \sum_{j=1}^{X} \mathbb{P}\big(\vec{y}_3 = \vec{e}_n, \vec{r}_3 = \vec{e}_m, a_3 = k | x_2 = i, a_2 = l, x_3 = j\big) \mathbb{P}\big(x_3 = j | x_2 = i, a_2 = l\big) \\
&= \sum_{j=1}^{X} \mathbb{P}\big(\vec{r}_3 = \vec{e}_m | x_3 = j, a_3 = k\big) \mathbb{P}(a_3 = k | \vec{y}_3 = \vec{e}_n) \mathbb{P}\big(\vec{y}_3 = \vec{e}_n | x_3 = j\big) \mathbb{P}\big(x_3 = j | x_2 = i, a_2 = l\big) \\
&= f_\pi(k | \vec{e}_n) \sum_{j=1}^{X} f_R(\vec{e}_m | j, k) f_O(\vec{e}_n | j) f_T(j | i, l),
\end{aligned}
$$

where we factorized the three components in the definition of $V_3^{(l)}$ and used the graphical model of the POMDP to consider their dependencies. We introduce an auxiliary matrix $W \in \mathbb{R}^{d \times X}$ such that

$$[W]_{s,j} = [W]_{(n,m,k),j} = f_\pi(k|\vec{e}_n) f_R(\vec{e}_m|j,k) f_O(\vec{e}_n|j),$$

which contain all known values, and for any state i and action l we can restate the definition of the third view as

$$W[T]_{i,:,l} = [V_3^{(l)}]_{:,i}, \tag{5.20}$$

which allows computing the transition model as $[T]_{i,:,l} = W^\dagger [V_3^{(l)}]_{:,i}$, where $W^\dagger$ is the pseudo-inverse of W. While this change in the definition of the third view allows a significant relaxation of the original assumption, it comes at the cost of potentially worsening the bound on $\widehat{f}_T$ in Thm. 5.1. In fact, it can be shown that

$$\|\widetilde{f}_T(\cdot|i,l) - f_T(\cdot|i,l)\|_F \leq \mathcal{B}'_T := \max_{l'=1,\dots,A} \frac{C_T A Y R}{\lambda^{(l')}} \sqrt{\frac{X A \log(1/\delta)}{N(l')}}. \tag{5.21}$$

Beside the dependency on multiplication of Y, R, and R, which is due to the fact that now $V_3^{(l)}$ is a larger matrix, the bound for the transitions triggered by an action l scales with the number of samples from the least visited action. This is due to the fact that now the matrix W involves not only the action for which we are computing the transition model but all the other actions as well. As a result, if any of these actions is poorly visited, W cannot be accurately estimated is some of its parts and this may negatively affect the quality of estimation of the transition model itself. This directly propagates to the regret analysis, since now we require all the actions to be repeatedly visited enough. The immediate effect is the introduction of a different notion of diameter. Let $\tau_{M,\pi}^{(l)}$ the mean passage time between

two steps where action l is chosen according to policy $\pi \in \mathcal{P}$, we define

$$D_{\text{ratio}} = \max_{\pi \in \mathcal{P}} \frac{\max_{l \in \mathcal{A}} \tau_{M,\pi}^{(l)}}{\min_{l \in \mathcal{A}} \tau_{M,\pi}^{(l)}} \tag{5.22}$$

as the diameter ratio, which defines the ratio between maximum mean passing time between choosing an action and choosing it again, over its minimum. As it mentioned above, in order to have an accurate estimate of f_T all actions need to be repeatedly explored. The D_{ratio} is small when each action is executed frequently enough and it is large when there is at least one action that is executed not as many as others. Finally, we obtain

$$\text{Reg}_N \leq \widetilde{O}\Big(\frac{r_{\max}}{\overline{\lambda}} \sqrt{Y R D_{\text{ratio}} N \log N} X^{3/2} A(D+1)\Big).$$

While at first sight this bound is clearly worse than in the case of stronger assumptions, notice that $\overline{\lambda}$ now contains the smallest singular values of the newly defined views. In particular, as $V_3^{(l)}$ is larger, also the covariance matrices $K_{\nu,\nu'}$ are bigger and have larger singular values, which could significantly alleviate the inverse dependency on $\overline{\sigma}_{1,2}$ and $\overline{\sigma}_{2,3}$. As a result, relaxing Asm. 4 may not necessarily worsen the final bound since the bigger diameter may be compensated by better dependencies on other terms. We leave a more complete comparison of the two configurations (with or without Asm. 4) for future work.

5.5 Experiments

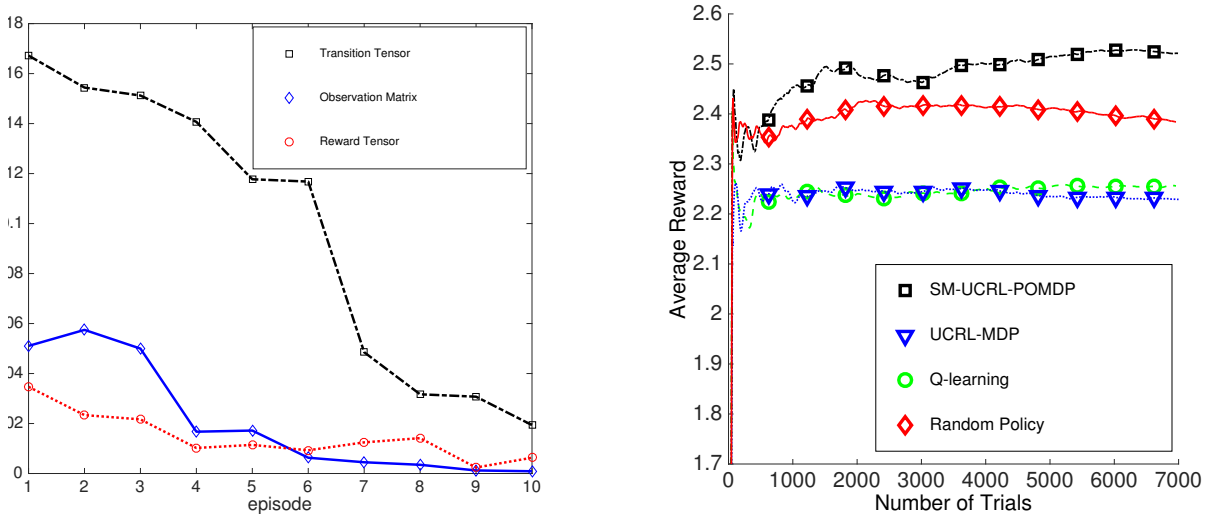
Here, we illustrate the performance of our method on a simple synthetic environment which follows a POMDP structure with $X = 2$, $Y = 4$, $A = 2$, $R = 4$, and $r_{max} = 4$. We find that spectral learning method converges quickly to the true model parameters, as seen in Fig. [5.2(a)]. Estimation of the transition tensor T takes longer compared to estimation of observation matrix O and reward Tensor R. This is because the observation and reward

matrices are first estimated through tensor decomposition, and the transition tensor is estimated subsequently through additional manipulations. Moreover, the transition tensor has more parameters since it is a tensor (involving observed, hidden and action states) while the observation and reward matrices involve fewer parameters.

For planning, given the POMDP model parameters, we find the memoryless policy using a simple alternating minimization heuristic, which alternates between updates of the policy and the stationary distribution. We find that in practice this converge to a good solution. The regret bounds are shown in Fig. [5.2(b)]. We compare against the following policies: (1) baseline random policies which simply selects random actions without looking at the observed data, (2) UCRL-MDP Auer et al. (2009) which attempts to fit a MDP model to the observed data and runs the UCRL policy, and (3) Q-Learning Watkins and Dayan (1992b) which is a model-free method that updates policy based on the Q-function. We find that our method converges much faster than the competing methods. Moreover, it converges to a much better policy. Note that the MDP-based policies UCRL-MDP and Q-Learning perform very poorly, and are even worse than the baseline are too far from SM-UCRL policy. This is because the MDP policies try to fit data in high dimensional observed space, and therefore, have poor convergence rates. On the other hand, our spectral method efficiently finds the correct low dimensional hidden space quickly and therefore, is able to converge to an efficient policy.

5.6 Conclusion

We introduced a novel RL algorithm for POMDPs which relies on a spectral method to consistently identify the parameters of the POMDP and an optimistic approach for the solution of the exploration–exploitation problem. For the resulting algorithm we derive confidence intervals on the parameters and a minimax optimal bound for the regret.



a)Learning by Spectral Method

b) Regret Performance

Figure 5.2: (a)Accuracy of estimated model parameter through tensor decomposition. See h Eqs. 5.11,5.10 and 5.12. (b)Comparison of SM-UCRL-POMDP is our method, UCRL-MDP which attempts to fit a MDP model under UCRL policy, $\epsilon - greedy$ Q-Learning, and a Random Policy.

This work opens several interesting directions for future development. **1)** SM-UCRL cannot accumulate samples over episodes since Thm. 5.1 requires samples to be drawn from a fixed policy. While this does not have a very negative impact on the regret bound, it is an open question how to apply the spectral method to all samples together and still preserve its theoretical guarantees. **2)** While memoryless policies may perform well in some domains, it is important to extend the current approach to bounded-memory policies. **3)** The POMDP is a special case of the predictive state representation (PSR) model Littman et al. (2001), which allows representing more sophisticated dynamical systems. Given the spectral method developed in this paper, a natural extension is to apply it to the more general PSR model and integrate it with an exploration–exploitation algorithm to achieve bounded regret.

Algorithm 6 Estimation of the POMDP parameters. The routine TensorDecomposition refers to the spectral tensor decomposition method of Anandkumar et al. (2012).

Input:

Policy density f_π, number of states X

Trajectory $\langle(\vec{y}_1, a_1, \vec{r}_1), (\vec{y}_2, a_2, \vec{r}_2), \ldots, (\vec{y}_N, a_N, \vec{r}_N)\rangle$

Variables:

Estimated second and third views $\widehat{V}_2^{(l)}$, and $\widehat{V}_3^{(l)}$ for any action $l \in [A]$

Estimated observation, reward, and transition models $\widehat{f}_O$, $\widehat{f}_R$, $\widehat{f}_T$

for $l = 1, \ldots, A$ **do**

Set $\mathcal{T}(l) = \{t \in [N-1] : a_t = l\}$ and $N(l) = |\mathcal{T}(l)|$

Construct views $\vec{v}_{1,t}^{(l)} = (a_{t-1}, \vec{y}_{t-1}, \vec{r}_{t-1})$, $\vec{v}_{2,t}^{(l)} = (\vec{y}_t, \vec{r}_t)$, $\vec{v}_{3,t}^{(l)} = \vec{y}_{t+1}$ for any $t \in \mathcal{T}(l)$

Compute covariance matrices $\widehat{K}_{3,1}^{(l)}$, $\widehat{K}_{2,1}^{(l)}$, $\widehat{K}_{3,2}^{(l)}$ as

$$\widehat{K}_{\nu,\nu'}^{(l)} = \frac{1}{N(l)} \sum_{t \in \mathcal{T}(l)} \vec{v}_{\nu,t}^{(l)} \otimes \vec{v}_{\nu',t}^{(l)}; \;\; \nu, \nu' \in \{1,2,3\}$$

Compute modified views $\widetilde{\vec{v}}_{1,t}^{(l)} := \widehat{K}_{3,2}^{(l)} (\widehat{K}_{1,2}^{(l)})^\dagger \vec{v}_1$, $\widetilde{\vec{v}}_{2,t}^{(l)} := \widehat{K}_{3,1}^{(l)} (\widehat{K}_{2,1}^{(l)})^\dagger \vec{v}_{2,t}^{(l)}$ for any $t \in \mathcal{T}(l)$

Compute second and third moments

$$\widehat{M}_2^{(l)} = \frac{1}{N(l)} \sum_{t \in \mathcal{T}_l} \widetilde{\vec{v}}_{1,t}^{(l)} \otimes \widetilde{\vec{v}}_{2,t}^{(l)}, \quad \widehat{M}_3^{(l)} = \frac{1}{N(l)} \sum_{t \in \mathcal{T}_l} \widetilde{\vec{v}}_{1,t}^{(l)} \otimes \widetilde{\vec{v}}_{2,t}^{(l)} \otimes \vec{v}_{3,t}^{(l)}$$

Compute $\widehat{V}_3^{(l)} = \textsc{TensorDecomposition}(\widehat{M}_2^{(l)}, \widehat{M}_3^{(l)})$

Compute $\widehat{\mu}_{2,i}^{(l)} = \widehat{K}_{1,2}^{(l)} (\widehat{K}_{3,2}^{(l)})^\dagger \widehat{\mu}_{3,i}^{(l)}$ for any $i \in [X]$

Compute $\widehat{f}(\vec{e}_m | i, l) = \sum_{n'=1}^{Y} [\widehat{V}_2^{(l)}]_{(n',m),i}$ for any $i \in [X]$, $m \in [R]$

Compute $\rho(i, l) = \sum_{m'=1}^{R} \sum_{n'=1}^{Y} \frac{[V_2^{(l)}]_{(n',m'),i}}{f_\pi(l|\vec{e}_{n'})}$ for any $i \in [X]$, $n \in [Y]$

Compute $\widehat{f}_O^{(l)}(\vec{e}_n | i) = \sum_{m'=1}^{R} \frac{[V_2^{(l)}]_{(n,m'),i}}{f_\pi(l|\vec{e}_n)\rho(i,l)}$ for any $i \in [X]$, $n \in [Y]$

end for

Compute bounds $\mathcal{B}_O^{(l)}$

Set $l^* = \arg\min_l \mathcal{B}_O^{(l)}$, $\widehat{f}_O = \widehat{f}_O^{l^*}$ and construct matrix $[\widehat{O}]_{n,j} = \widehat{f}_O(\vec{e}_n | j)$

Reorder columns of matrices $\widehat{V}_2^{(l)}$ and $\widehat{V}_3^{(l)}$ such that matrix $O^{(l)}$ and $O^{(l^*)}$ match, $\forall l \in [A]$[4]

for $i \in [X]$, $l \in [A]$ **do**

Compute $[T]_{i,:,l} = \widehat{O}^\dagger [\widehat{V}_3^{(l)}]_{:,i}$

end for

Return: $\widehat{f}_R$, $\widehat{f}_T$, $\widehat{f}_O$, $\mathcal{B}_R$, $\mathcal{B}_T$, $\mathcal{B}_O$

Algorithm 7 The SM-UCRL algorithm.

Input: Confidence δ'

Variables:

Number of samples $N^{(k)}(l)$

Estimated observation, reward, and transition models $\widehat{f}_O^{(k)}$, $\widehat{f}_R^{(k)}$, $\widehat{f}_T^{(k)}$

Initialize: $t = 1$, initial state x_1, $\delta = \delta'/N^6$, $k = 1$

while $t < N$ **do**

Compute the estimated POMDP $\widehat{M}^{(k)}$ with the Alg. 6 using $N^{(k)}(l)$ samples per action

Compute the set of admissible POMDPs $\mathcal{M}^{(k)}$ using bounds in Thm. 5.1

Compute the optimistic policy $\widetilde{\pi}^{(k)} = \arg\max_{\pi \in \mathcal{P}} \max_{M \in \mathcal{M}^{(k)}} \eta(\pi; M)$

Set $v^{(k)}(l) = 0$ for all actions $l \in [A]$

while $\forall l \in [A], v^{(k)}(l) < 2N^{(k)}(l)$ **do**

Execute $a_t \sim f_{\widetilde{\pi}^{(k)}}(\cdot|\vec{y}_t)$

Obtain reward $\vec{r}_t$, observe next observation $\vec{y}_{t+1}$, and set $t = t + 1$

end while

Store $N^{(k+1)}(l) = \max_{k' \leq k} v^{(k')}(l)$ samples for each action $l \in [A]$

Set $k = k + 1$

end while

Chapter 6

Policy Gradient in Partially Observable MDPs

Trust Region Policy Optimization for POMDPs We propose Generalized Trust Region Policy Optimization (GTRPO), a policy gradient Reinforcement Learning (RL) algorithm for both Markov decision processes (MDP) and Partially Observable Markov Decision Processes (POMDP). Policy gradient is a class of model-free RL methods. Previous policy gradient methods are guaranteed to converge only when the underlying model is an MDP and the policy is run for an infinite horizon. We relax these assumptions to episodic settings and to partially observable models with memoryless policies. For the latter class, GTRPO uses a variant of the Q-function with only three consecutive observations for each policy updates, and hence, is computationally efficient. We theoretically show that the policy updates in GTRPO monotonically improve the expected cumulative return and hence, GTRPO has convergence guarantees.

6.1 Introduction

One of the central challenges in reinforcement learning is the design of efficient algorithms for high-dimensional environments. Recently, Deep-Q networks (Mnih et al., 2015) and its variants, as value-based model-free methods, have shown promise in scaling to large observational spaces. However, these methods are limited to MDPs and mainly dedicated to finite action spaces. Policy gradient methods (Aleksandrov et al., 1968) are another class of model-free methods with no model assumption, therefore conventional approach for continuous high-dimensional action spaces and more importantly for partially observable environments.

Policy gradient approaches mainly deploy Monte Carlo sampling for the gradient update but suffer from high variance gradient estimation (Rubinstein, 1969). To mitigate the high variance shortcoming, recent works deploy value-based methods to the gradient estimation and provide low variance policy gradient methods (Schulman et al., 2015; Lillicrap et al., 2015). However, they mainly assume the underlying environment is a Markov decision process (MDP), the policy is run to the infinite horizon, and the rewards are discounted (Schulman et al., 2015). In practice, many of these assumptions do not hold. The real-world problems are mainly partially observable, episodic and sometimes, the rewards are undiscounted. It is worth noting that even the empirical study provided in these previous works are episodic, while the theory assumes infinite horizon. If the underlying model of the environment is POMDP, applying MDP based method might result in policies with arbitrarily bad expected returns (Azizzadenesheli et al., 2017).

Table 6.1 categorizes the majority of RL problems concerning their observability level, policy class, horizon length, and discount factor. Prior methods mainly focus on the memory-less, infinite horizon, undiscounted MDPs. In this work, we focus on episodic MDPs and POMDP with memoryless policies. We investigate both discounted and undiscounted reward settings.

Table 6.1: Category of most RL problems

Observability	Policy Class	Horizon	Discounting
MDP	Memory-less	Infinite	Discounted
POMDP	Memory dependent	Episodic	Undiscounted

Generally, on-policy policy gradient methods collect data under the policies at hand (current policy) and exploit the acquired data to search for a new and potentially a better policy to deploy. The hope is that this procedure iteratively reinforces the algorithm's behavior and improves its expected return. Therefore, one of the central goals in policy gradient methods is to develop low variance policy updates which result in the monotonic improvements of the expected returns: the so-called Monotonic Improvement guarantee. Under the infinite horizon undiscounted setting with MDP modeling assumption, Kakade and Langford (2002); Schulman et al. (2015) study the trust-region methods, e.g., TRPO, a class of policy gradients methods which perform the policy search in the trust region around the current policy. They construct a surrogate objective using advantage functions and propose a low variance policy gradient updates. They prove that their low variance policy updates monotonically improves the expected return.

In the low sample setting, the accurate estimation of the trust regions is not tractable. TRPO requires to explicitly estimate the trust region to constrain the parameter space which may be hard to maintain in high dimensional and low samples settings. To mitigate this limitation, (Schulman et al., 2017) offer Proximal Policy Optimization (PPO), a simple extension to TRPO, which approximately retains the trust-region constraints directly on the policy space than the parameter space. It also significantly reduces the computation cost of TRPO, therefore it is a reasonable choice for empirical study.

Contributions: In this work, we extend the trust region methods, e.g., TRPO, PPO, to episodic MDPs. We show that deploying infinite horizon methods for episodic problem results in a biased estimation of the trust region. We show that it is necessary to incorporate the length of each episode to construct the trust region and extend TRPO and PPO to

episodic MDPs.

In presence of discount factor, it is intuitive that the later parts of an episode would have less contribution towards constructing the trust region than the earlier parts. This is also not captured in previous trust region works, e.g. TRPO, PPO. We further extend our analysis and introduce a new notion of distribution divergence, as a discounted sum of Kullback–Leibler (KL) divergences, and show how to construct trust regions in discounted reward settings.

Mainly, the optimal policies for MDPs are deterministic and memoryless. In contrast, we might consider a class of history-dependent policies when we deal with POMDPs. However, tackling history dependent policies can be computationally undecidable (Madani et al., 1999) or PSPACE-Complete (Papadimitriou and Tsitsiklis, 1987b), and hence, many works consider the class of stochastic memoryless policies (Azizzadenesheli et al., 2016c). In this work, to avoid the computation intractability of history dependent policies, we focus on the class of memory-less policies. Generally, the optimal policies of POMDPs in the class of memoryless policies are indeed stochastic. It is also worth noting that extending the value-based methods through Bellman equations to stochastic memoryless or limited memory policies is not possible if optimality is concerned.

Despite the MDP assumption in the mainstream policy gradient works, empirical studies have demonstrated superior performance when the classes of stochastic policies are considered, e.g., TRPO. The stochastic policies are also known to contribute to the exploration. Many policy gradient algorithms mainly do not converge to deterministic policies in the evaluation period, which is another piece of evidence on partial observability of the environments. Moreover, Sutton et al. (1998) argues that when function approximation methods are deployed to represent the states, due to loss of information in the representation function, the problem inherently is a POMDP.

We propose GTRPO, a policy gradient algorithm for episodic POMDPs. GTRPO deploys three consecutive observations in order to approximate an advantage function and computes a low variance estimation of the policy gradient. Surprisingly, deploying three consecutive observations matches the statement in (Azizzadenesheli et al., 2016c) which shows statistics of three consecutive observations are necessary to learn the POMDP dynamics and guarantee a regret upper bound in the model-based RL. We construct a trust region for the policy search in GTRPO and show that the policy updates are guaranteed to monotonically improve the expected return. To the best of our knowledge, GTRPO is the first algorithm with monotonic improvement guarantee for the class of POMDP problems.

For the experimental study of GTRPO, we deploy the same methodology used in PPO to reduce the computation cost in TRPO. We apply GTRPO on a variety of RoboSchool (Schulman et al., 2017) environments, which are the extension to the MuJoCo environments (Todorov et al., 2012). We empirically show that despite the computational complexity introduced by most of POMDP based methods, computation complexity introduced by GTRPO is in the same order as its MDP based predecessors. We study GTRPO performance on these simulated environments (RoboSchool environments are almost MDPs, and we do not aim to outperform MDP based methods in these experiments) and report its behavior under different simulation design choices. Throughout the experiments, we observe a similar behavior of the MDP based approach PPO and POMDP based approach GTRPO.

6.2 Preliminaries

An episodic POMDP M is a tuple $\langle \mathcal{X}, \mathcal{A}, \mathcal{Y}, P_0, T, R, O, \gamma, x_T \rangle$ with latent state space $\mathcal{X}$, observation space $\mathcal{Y}$, action space $\mathcal{A}$, discount factor of $0 \leq \gamma \leq 1$ and stochastic reward distribution of $R(x, a)$ with mean $\overline{R}(x, a) = \mathbb{E}[R(x, a)],\ \forall x \in \mathcal{X}, a \in \mathcal{A}$. Let x_T denote the terminal state which is *accessible* from any other state, i.e. starting from any other state

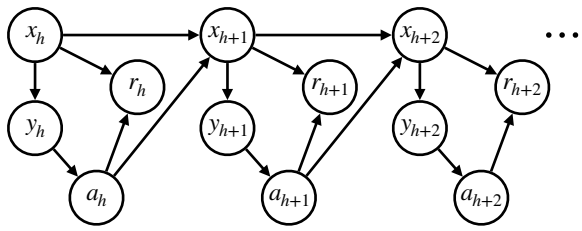


Figure 6.1: POMDP under a memory-less policy

there is a nonzero probability of reaching the x_T in finite time steps. The episode terminates when the process reaches x_T. The initial latent states are drawn from distribution P_1. The state dynamics follows transition density $T(x'|x,a),\ \forall x, x' \in \mathcal{X}, a \in \mathcal{A}$ and the observation process is generated using density $O(y|x),\ \forall x \in \mathcal{X}, y \in \mathcal{Y}$ where a memory-less policy is deployed Fig. 6.1.

We consider a set of continuously differentiable parameterized memory-less policies π_θ with $\theta \in \Theta$. For each y, a pair, $\pi_\theta(a|y)$ denotes the conditional probability distribution of choosing action a under the policy π_θ when an observation y is made. Furthermore, we define a random trajectory τ as a finite length $|\tau|$ sequence of events $\{(x_1, y_1, a_1, r_1), (x_2, y_2, a_2, r_2), \ldots (x_{|\tau|}, y_{|\tau|}, a_{|\tau|}, r_{|\tau|})\}$ where the termination happens at the step after $x_{|\tau|}$, i.e. $x_{|\tau|+1} = x_T$. Let $f(\tau;\theta),\ \forall \tau \in \Upsilon$ denote the probability distribution of trajectories under policy π_θ and Υ is the set of all possible trajectories. Furthermore, $R(\tau)$ denotes the cumulative γ-discounted rewards of the trajectory $\tau \in \Upsilon$. The agent goal is to maximize the unnormalized expected cumulative return $\eta(\theta) = \mathbb{E}_{\tau|\theta}[R(\tau)]$;

$$\theta^* = \arg\min_{\theta \in \Theta} \eta(\theta) := \int_{\tau \in \Upsilon} f(\tau;\theta) R(\tau) d\tau \tag{6.1}$$

with $\pi^* = \pi(\theta^*)$ the optimal policy.

6.3 Policy Gradient

In this section, we study the policy gradients methods for POMDPs. Generally, the optimization problem in Eq. 6.1 is a non-convex problem. Therefore, hill climbing methods such as gradient ascent based approaches might converge to the first order stationary points. Gradient ascent for Eq. 6.1 results in the policy gradient method. The policy gradient lemma states that the gradient of the expected cumulative return does not require the explicit knowledge of the dynamics but just the cumulative reward distribution (Rubinstein, 1969; Williams, 1992; Baxter and Bartlett, 2001b). This lemma has mainly been proven through the construction of score function (see the Appendix of Azizzadenesheli et al. (2018a)). In this section, we re-derive the same Lemma but through importance sampling since it is more related to the latter parts of this paper.

Importance sampling is a general technique for estimating the properties of a particular distribution, while only having samples generated from another distribution. One can estimate $\eta(\theta')$, $\theta' \in \Theta$, while the expectation is over the distribution induced by π_θ;

$$\begin{aligned}\eta(\theta') &= \mathbb{E}_{\tau|\theta'}\left[R(\tau)\right] = \int_{\tau\in\Upsilon} f(\tau;\theta)\left(\frac{f(\tau;\theta')}{f(\tau;\theta)}R(\tau)\right)d\tau \\ &= \mathbb{E}_{\tau|\theta}\left[\frac{f(\tau;\theta')}{f(\tau;\theta)}R(\tau)\right] \end{aligned} \tag{6.2}$$

as long as for each τ that $f(\tau;\theta') > 0$ also $f(\tau;\theta) > 0$. The gradient of $\eta(\theta')$ with respect to θ' is

$$\begin{aligned}\nabla_{\theta'}\eta(\theta') &= \mathbb{E}_{\tau|\theta}\left[\frac{\nabla_{\theta'} f(\tau;\theta')}{f(\tau;\theta)}R(\tau)\right] \\ &= \mathbb{E}_{\tau|\theta}[\frac{f(\tau;\theta')}{f(\tau;\theta)}\nabla_{\theta'}\log(f(\tau;\theta'))R(\tau)]\end{aligned}$$

The gradient at $\theta' = \theta$ is;

$$\nabla_{\theta'}\eta(\theta') \mid_{\theta'=\theta} = \mathbb{E}_{\tau|\theta}\left[\nabla_\theta \log(f(\tau;\theta))R(\tau)\right] \tag{6.3}$$

Since for each trajectory τ, the $\log(f(\tau;\theta))$;

$$\log\Big(P_1(x_1)O(y_1|x_1)R(r_1|x_1,a_1)\Pi_{h=2}^{|\tau|}T(x_h|x_{h-1},a_{h-1}) \\ O(y_h|x_h)R(r_h|x_h,a_h)\Big) + \log\Big(\Pi_{h=1}^{|\tau|}\pi_\theta(a_h|y_h)\Big)$$

and the first part is independent θ we have;

$$\nabla_\theta \log(f(\tau;\theta)) = \nabla_\theta \log\Big(\Pi_{h=1}^{|\tau|}\pi_\theta(a_h|y_h)\Big)$$

This derivation suggest that given trajectories under a policy π_θ we can compute the gradient of the expected return with respect to the parameters of π_θ without the knowledge of the dynamics. In practice, however we are not able to compute the exact expectation. Instead we can deploy Monte Carlo sampling technique to estimate the gradient. Given m trajectories $\{\tau^1,\ldots,\tau^m\}$ with elements $(x_h^t, y_h^t, a_h^t, r_h^t), \forall h \in \{1,\ldots,|\tau^t|\}$ and $\forall t \in \{1,\ldots,m\}$ generated under a policy π_θ, we can estimate the gradient in Eq. 6.3 at point θ;

$$\nabla_\theta \widehat{\eta}(\theta) = \frac{1}{m}\sum_{t=1}^{m}\nabla_\theta \log\Big(\Pi_{h=1}^{|\tau^t|}\pi_\theta(a_h^t|y_h^t)\Big)R(\tau^t) \tag{6.4}$$

which returns a high variance estimation of the gradient.

6.3.1 Natural Policy Gradient

Generally, the notion of gradient depends on the parameter metric space. Given a pre-specified Riemannian metric, a gradient direction is defined. When the metric is Euclidean,

the notion of gradient reduces to the standard gradient (Lee, 2006). This general notion of gradient adjusts the standard gradient direction based on the local curvature induced by the Riemannian manifold of interest. Valuable knowledge of the curvature assists to find an ascent direction which might conclude to big ascend in the objective function. This approach is also interpreted as a trust region method where we are interested in assuring that the ascent steps do not change the objective beyond a safe region where the local curvature might not stay valid. In general, a valuable manifold might not be given, and we need to adopt one. Fortunately, when the objective function is an expectation over a parameterized distribution, Amari (2016) recommends employing a Riemannian metric, induced by the Fisher information. This choice of metric results in a well knows notion of gradient, so-called *natural gradient*. For the objective function in 6.1, the Fisher information matrix is defined as follows;

$$F(\theta) := \int_{\tau \in \Upsilon} f(\tau;\theta) \Big[\nabla_\theta \log\left(f(\tau;\theta)\right) \nabla_\theta \log\left(f(\tau;\theta)\right)^\top \Big] d\tau \tag{6.5}$$

Natural gradients are firstly deployed by Kakade (2002) for RL in MDPs. Consequently, the direction of the gradient with respect to F is defined as $F(\theta)^{-1}\nabla_\theta(\eta(\theta))$. One can compute the inverse of this matrix to come up with the direction of the natural gradient. Since neither storing the Fisher matrix is always possible nor computing the inverse is practical, direct utilization of $F(\theta)^{-1}\nabla_\theta(\eta(\theta))$ is not feasible. Similar to the approach in TRPO, we suggest to first deploy $\mathcal{D}_{KL}$ divergence substitution technique and then conjugate gradient method to tackle the computation and storage bottlenecks.

Lemma 3. *Under some regularity conditions;*

$$\nabla^2_{\theta'} \mathcal{D}_{KL}(\theta, \theta')|_{\theta'=\theta} = F(\theta) \tag{6.6}$$

with $\mathcal{D}_{KL}(\theta, \theta') := -\int_{\tau \in \Upsilon} f(\tau;\theta) \log\left(f(\tau;\theta')/f(\tau;\theta)\right) d\tau$

The Lemma 3 is a known lemma in the literature and we provide its proof in the Appendix of Azizzadenesheli et al. (2018a). In practice, it is not feasible to compute the expectation in neither the Fisher information matrix nor in the $\mathcal{D}_{KL}$ divergence, but rather their empirical estimates. Given m trajectories

$$\begin{aligned}
&\nabla^2_{\theta'}\widehat{\mathcal{D}}_{KL}(\theta,\theta')|_{\theta'=\theta} \\
&\quad = -\frac{1}{m}\nabla^2_{\theta'}\sum_{t=1}^{m}\Big[\log\Big(\Pi_{h=1}^{|\tau^t|}\pi_{\theta'}(a_h^t|y_h^t)\Big)\log\Big(\Pi_{h=1}^{|\tau^t|}\pi_{\theta}(a_h^t|y_h^t)\Big)\Big]|_{\theta'=\theta} \\
&\quad = -\frac{1}{m}\nabla^2_{\theta'}\sum_{t=1}^{m}\sum_{h=1}^{|\tau^t|}\log\left(\frac{\pi_{\theta'}(a_h^t|y_h^t)}{\pi_{\theta}(a_h^t|y_h^t)}\right)
\end{aligned}$$

This derivation of $\mathcal{D}_{KL}$ is common between MDPs and POMDPs. The analysis in most of the state-of-the-art policy gradient methods, e.g. TRPO, PPO, are dedicated to infinite horizon MDPs, while almost all the experimental studies are in the episodic settings. Therefore the estimator used in these methods;

$$\nabla^2_{\theta'}\widehat{\mathcal{D}}^{TRPO}_{KL}(\theta,\theta')|_{\theta'=\theta} = -\frac{1}{\sum_t^m|\tau^t|}\nabla^2_{\theta'}\sum_{t=1}^{m}\sum_{h=1}^{|\tau^t|}\log\left(\frac{\pi_{\theta'}(a_h^t|y_h^t)}{\pi_{\theta}(a_h^t|y_h^t)}\right)$$

is a bias estimation of the $\mathcal{D}_{KL}$ in episodic settings.

6.3.2 $\mathcal{D}_{KL}$ vs ${\mathcal{D}_{KL}}^{TRPO}$

The use of $\mathcal{D}_{KL}$ instead of ${\mathcal{D}_{KL}}^{TRPO}$ is motivated by theory and also intuitively recommended. A small change in the policy at the beginning of a short episodes does not make a drastic shift in the distribution of the trajectory but might cause radical shifts when the trajectory length is long. Therefore, for longer horizons, the trust region needs to shrink. Consider two trajectories, one long and one short. The $\mathcal{D}_{KL} \leq \delta$ induces a region which allows higher changes in the policy for short trajectory while limiting changes in long trajec-

tory. While ${\mathcal{D}_{KL}}^{TRPO} \leq \delta$ induces the region which does not convey the length of trajectories and looks at each sample as it experienced in a stationary distribution of an infinite horizon MDP.

Consider a toy RL problem where at the beginning of the learning, when the policy is not good, the agent dies at early stages of the episodes (termination). In this case, the trust region under $\mathcal{D}_{KL}$ is vast and allows for substantial change in the policy space, while again ${\mathcal{D}_{KL}}^{TRPO}$ does not consider the length of the episode. On the other hand, toward the end of the learning, when the agent has leart a good policy, the length of the horizon grows, and small changes in the policy might cause drastic changes in the trajectory distribution. Therefore the trust region shrinks again, and just a small changes in the policy space are allowed, which is again captured by $\mathcal{D}_{KL}$ but not by ${\mathcal{D}_{KL}}^{TRPO}$.

Compatible Function Approximation As it is mentioned before, one way of computing the direction of the natural gradient is to estimate the $\widehat{\mathcal{D}}_{KL}$ and use conjugate gradient methods to find $F^{-1}\nabla_\theta(\eta)$. There is also an alternative way to estimate $F^{-1}\nabla_\theta(\eta)$, which is based on compatible function approximation methods. Kakade (2002) study this approach in the context of MDPs. In the following, we develop this approach for POMDPs. Consider a feature map $\phi(\tau)$ in a desired ambient space defined on Γ. We approximate the return $R(\tau)$ via a linear function ω on the feature representation $\phi(\tau)$, i.e.,

$$\min_{\omega} \epsilon(\omega) \ s.t.; \ \epsilon(\omega) := \int_{\tau\in\Upsilon} f(\tau,\theta)[\phi(\tau)^\top\omega - R(\tau)]^2 d\tau$$

To find the optimal ω we take the gradient of $\epsilon(\omega)$ and set it to zero;

$$0 = \nabla_\omega \epsilon(\omega)|_{\omega=\omega^*} = \int_{\tau\in\Upsilon} 2f(\tau,\theta)\phi(\tau)[\phi(\tau)^\top\omega^* - R(\tau)]d\tau$$

For the optimality,

$$\int_{\tau\in\Upsilon} f(\tau,\theta)\phi(\tau)\phi(\tau)^\top\omega^* d\tau = \int_{\tau\in\Upsilon} f(\tau,\theta)\phi(\tau;\theta)R(\tau)d\tau$$

If we consider the $\phi(\tau) = \nabla_\theta \log\left(\Pi_{h=1}^{|\tau|}\pi_\theta(a_h|y_h)\right)$, the LHS of this equation is $F(\theta)\omega^*$. Therefore

$$F(\theta)\omega = \nabla_\theta\eta(\theta) \Longrightarrow \omega^* = F(\theta)^{-1}\nabla\rho$$

In practice, depending on the problem at hand, either of the discussed approaches of computing the natural gradient might be applicable. Due to the close relationship between $\mathcal{D}_{KL}$ and Fisher information matrix Lemma 3 and also the fact that the Fisher matrix is equal to second order Taylor expansion of $\mathcal{D}_{KL}$, instead of considering the area $\| (\theta-\theta')^\top F (\theta-\theta') \|_2 \leq \delta$, or $\| (\theta-\theta')^\top \nabla^2_{\theta'}\mathcal{D}_{KL}(\theta,\theta')|_{\theta'=\theta} (\theta-\theta') \|_2 \leq \delta$ for construction of the trust region, we can approximately consider $\mathcal{D}_{KL}(\theta,\theta') \leq \delta/2$. This relationship between these three approaches in constructing the trust region is used throughout this paper.

6.4 TRPO for POMDPS

In this section we extend the MDP analysis in Kakade and Langford (2002); Schulman et al. (2015) to POMDPs, propose GTRPO, and derive a guarantee on its monotonic improvement property. We prove the monotonic improvement property using $\mathcal{D}_{KL}$. Moreover, we propose a discount factor dependent divergence and provide the monotonic improvement guarantee w.r.t. this new divergence.

The $\mathcal{D}_{KL}$ divergence and Fisher information matrix in Eq. 6.6, Eq. 6.5 do not convey the effect of the discount factor. Consider a setting with a small discount factor γ. In this setting,

we do not mind drastic distribution changes in the latter part of episodes. Therefore, we desire to have a even wider trust region and allow bigger changes for later parts of the trajectories. This is a valid intuition and in the following, we re-derive the $\mathcal{D}_{KL}$ divergence by also incorporating γ. Let $\tau_{h'}^{h}$ denote the elements in τ from the time step h' up to the time step h; we rewrite $\eta(\theta)$ as follows;

$$\eta(\theta) = \int_{\tau \in \Upsilon} f(\tau;\theta) R(\tau) d\tau = \int_{\tau \in \Upsilon} \sum_{h=1}^{|\tau|} f(\tau_1^h;\theta) \gamma^h r_h(\tau) d\tau$$

Following the Amari (2016) reasoning for Fisher information of each component of the sum, we derive a γ dependent divergence;

$$\mathcal{D}_\gamma(\pi_\theta, \pi_{\theta'}) = \sum_{h=1}^{\tau_{\max}} \gamma^h \mathcal{D}_{KL}\big(\tau_1^h \sim f(\cdot;\pi_{\theta'}), \tau_1^h \sim f(\cdot;\pi_\theta)\big) \tag{6.7}$$

For some upper bound on the trajectory lengths, $\tau_{\max}$. This divergence less penalizes the distribution mismatch in the later part of trajectories. Similarly, taking into account the relationship between KL divergence and Fisher information we have discount factor γ dependent definition of the Fisher information;

$$F_\gamma(\theta) := \int_{\tau \in \Upsilon} \sum_{h=1}^{\tau_{\max}} \gamma^h f(\tau_1^h;\theta) \left[\nabla_\theta \log\big(f(\tau_1^h;\theta)\big) \nabla_\theta \log\big(f(\tau_1^h;\theta)\big)^\top \right] d\tau$$

In the following we develop GTRPO monotonic improvement guarantee under both $\mathcal{D}_\gamma$ and $\mathcal{D}_{KL}$.

6.4.1 Advantage function on the hidden states

Let π_θ, the *current policy*, denote the policy under which we collect data, and $\pi_{\theta'}$, the *new policy*, the policy which we evaluate its performance. Generally, any memory-less policy on the observation space is transferable to a policy on the latent states as follows; $\pi(a|x) = \int_{y\in\mathcal{Y}} \pi(a|y)O(y|x)dy$ for each pair of (x, a). Consider the case where the agent also observes the latent state, i.e. POMDP →MDP. Since the dynamics on the latent states is MDP, we define the advantage function on the latent states. At time step h of an episode;

$$\widetilde{A}_\pi(a, x, h) =$$
$$\mathbb{E}_{x'\sim T(x'|x,a,h)}\left[r(x, a, h) + \gamma\widetilde{V}_\pi(x', h) - \widetilde{V}_\pi(x, h)\right]$$

Where $\widetilde{V}_\pi$ denote the value function of underlying MDP of latent states when a policy π is deployed. For this choice of advantage function we can write;

$$\begin{aligned}
&\mathbb{E}_{\tau\sim f(\tau,\pi_{\theta'})}\left[\sum_h^{|\tau|}\gamma^h\widetilde{A}_{\pi_\theta}(x_h, a_h, h)\right]\\
&= \mathbb{E}_{\tau\sim f(\tau,\pi_{\theta'})}\left[\sum_h^{|\tau|}\gamma^h\left[r(x_h, a_h, h) + \gamma\widetilde{V}_{\pi_\theta}(x_{h+1}, h) - \widetilde{V}_{\pi_\theta}(x_h, h)\right]\right]\\
&= \mathbb{E}_{\tau\sim f(\tau,\pi_{\theta'})}\left[\sum_h^{|\tau|}\gamma^h r_h\right] - \mathbb{E}_{x_0\sim P_1(x)}\left[\widetilde{V}_{\pi_\theta}(x_0)\right]\\
&= \eta(\pi_{\theta'}) - \eta(\pi_\theta)
\end{aligned}$$

This equality suggests that if we have the advantage function of the current policy π_θ and sampled trajectories from $\pi_{\theta'}$, we could compute and maximize the improvement in the expected return $\eta(\pi_{\theta'}) - \eta(\pi_\theta)$ or, potently, directly just maximize the expected return for $\pi_{\theta'}$ without incorporating π_θ. In practice, we do not have sampled trajectories from the new policy $\pi_{\theta'}$, rather we have sampled trajectories from the current policy π_θ. Therefore, we are interested in maximizing the following surrogate objective function since we can compute

it;

$$\widetilde{L}_{\pi_\theta}(\pi_{\theta'}) := \eta(\pi_\theta) + \mathbb{E}_{\tau\sim\pi_\theta, a'_h\sim\pi_{\theta'}(a'_h|x_h,h)}\left[\sum_{h}^{|\tau|}\gamma^h \widetilde{A}_{\pi_\theta}(x_h, a'_h, h)\right]$$

For infinite horizon MDPs when O is an identity map, i.e., $x_h = y_h$, Kakade and Langford (2002); Schulman et al. (2015) show that optimizing $\widetilde{L}_{\pi_\theta}(\pi_{\theta'})$ over θ' can provide an improvement in the expected discounted return. They derive a lower bound on this improvement if the $\mathcal{D}_{KL}$ between $\pi_{\theta'}$ and π_θ for all x's is bounded. In the following, we extend these analyses to the general class of environments, i.e. POMDPs and show such guarantees are conserved.

Generally, in POMDPs, when classes of memory-less policies are regarded, neither Q nor V functions are well-defined as they are for MDP through the Bellman optimality equations. In the following, we define two quantities similar to the Q and V in MDPs and for the simplicity we use the same Q and V notation for them. The conditional value and Q-value functions of POMDPs

$$\begin{aligned} V_\pi(y_h, h, y_{h-1}, a_{h-1}) &:= \mathbb{E}_\pi\left[\sum_{h'}^{H}\gamma^{h'} r_{h'}|y_h, y_{h-1}, a_{h-1}\right] \\ Q_\pi(y_{h+1}, a_h, y_h, h) &:= \mathbb{E}_\pi\left[\sum_{h'}^{H}\gamma^{h'} r_{h'}|y_h, y_{h+1}, a_h\right] \end{aligned} \tag{6.8}$$

For $h = 0$ we relax the conditioning on y_{h-1} for V_π and simply denote it as $V_\pi(y, 0)$. Deploying these two quantities, we define the advantage function as follows;

$$\begin{aligned} &A_\pi(y_{h+1}, a_h, y_h, h, y_{h-1}, a_{h-1}) \\ &= Q_\pi(y_{h+1}, a_h, y_h, h) - V_\pi(y_h, h, y_{h-1}, a_{h-1}) \end{aligned}$$

The relationship between these two quantity is as follows;

$$Q_\pi(y_{h+1}, a_h, y_h, h) :=$$
$$\mathbb{E}_\pi\left[r_h | y_{h+1}, a_h, y_h\right] + \gamma V_\pi(y_{h+1}, h+1, a_h, y_h)$$

Furthermore, we defined the following surrogate objective;

$$L_{\pi_\theta}(\pi_{\theta'}) = \eta(\pi_\theta) + \mathbb{E}_{\tau\sim\pi_\theta, a\sim\pi_{\theta'}(a|y)} \sum_{h}^{|\tau|} \gamma^h A_{\pi_\theta}(y_{h+1}, a_h, y_h, h, y_{h-1}, a_{h-1}) \tag{6.9}$$

Similar to MDPs, one can compute and maximize this surrogate objective function in Eq. 6.9 by just having sampled trajectories and advantage function of the current policy π_θ. But the domain of trust region for the policy search stays unknown. In the following section, we present the trust region for POMDPs.

Reward Structure: Similar to MDPs where the reward distribution given the current state, current action and the next state is conditionally independent of the rest of the events, we assume that the reward distribution given the current observation, current action and the next observation is conditionally independent of the rest of the events.

Under this structure we have;

Lemma 4. *The improvement in expected return, $\eta(\pi_{\theta'}) - \eta(\pi_\theta)$ is equal to;*

$$\mathbb{E}_{\tau\sim\pi_{\theta'}} \sum_{h}^{|\tau|} \gamma^h A_{\pi_\theta}(y_{h+1}, a_h, y_h, h, y_{h-1}, a_{h-1})$$

Proof of Lemma 4 in the Appendix of Azizzadenesheli et al. (2018a).

6.4.2 GTRPO

Algorithm 8 GTRPO

1: Initial π_{θ_0}, ϵ', δ'
2: Choice of divergence $\mathcal{D}$: $\mathcal{D}_{KL}$ or $\mathcal{D}_{\gamma}$
3: **for** episode = 1 until convergence **do**
4: Estimate the advantage function $\widehat{A}$
5: Construct the surrogate objective $\widehat{L}_{\pi_{\theta_{t-1}}}(\pi_\theta)$
6: Find the next policy

$$\pi_{\theta_t} = \arg\max_{\theta} L_{\pi_{\theta_{t-1}}}(\pi_\theta) \quad , s.t$$

$$\frac{1}{2}\|(\theta - \theta_{t-1})^{\top}\nabla^2_{\theta'}\mathcal{D}_{\gamma}(\theta_{t-1}, \theta')|_{\theta'=\theta_{t-1}}(\theta - \theta_{t-1})\|_2 \leq \delta'$$

7: **end for**

We propose generalized trust region policy optimization (GTRPO) as a policy gradient algorithm for POMDPs. GTRPO deploys its current policy to compute the advantage function and then maximize the advantage function over its actions in the vicinity of the current policy. This algorithm is almost identical to its predecessor TRPO except instead of maximizing over on observed hidden state dependent advantage function, $A_{\pi_\theta}(a_h, x_h, h)$, it maximizes over $A_{\pi_\theta}(y_{h+1}, a_h, y_h, h, y_{h-1}, a_{h-1})$ Alg. 8. It is important to note the one can easily turn the current implementations of TRPO to GTRPO by only changing the line of the code corresponding to the advantage function and substitute it with the proposed one. Moreover, if the model is MDP, $A_{\pi_\theta}(y_{h+1}, a_h, y_h, h, y_{h-1}, a_{h-1})$ is equivalent to $A_{\pi_\theta}(x_{h+1}, a_h, x_h)$ where after marginalizing out x_{h+1} in the expectation we end up with $A_{\pi_\theta}(a_h, x_h)$ and recover TRPO algorithm.

In practice, one can estimate the advantage function $A_{\pi_\theta}(y_{h+1}, y, a, h, y_{h-1}, a_{h-1})$ by approximating $Q_{\pi_\theta}(y_{h+1}, a, y_h, h)$ and $V_{\pi_\theta}(y_h, h, y_{h-1}, a_{h-1})$ using on-policy data of π_θ. Moreover, for $L_{\pi_\theta}(\pi_{\theta'})$ we have;

$$L_{\pi_\theta}(\pi_\theta) = \eta(\pi_\theta), \ \ and \ \nabla_{\theta'} L_{\pi_\theta}(\pi_{\theta'})|_{\pi_\theta=\pi_\theta} = \nabla_\theta \eta(\pi_\theta)$$

In the following we show that maximizing $L_{\pi_\theta}(\pi_{\theta'})$ over θ' results in a lower bound on the

improvement $\eta(\pi_{\theta'}) - \eta(\pi_\theta)$ when π_θ and $\pi_{\theta'}$ are close under $\mathcal{D}_{KL}$ or $\mathcal{D}_\gamma$ divergence. Lets define the averaged advantage function

$$\overline{A}_{\pi_\theta,\pi_{\theta'}}(y_{h+1}, y_h, h, y_{h-1}, a_{h-1}) = \\ \mathbb{E}_{a\sim\pi_{\theta'}}\left[A_{\pi_\theta}(y_{h+1}, a, y_h, h, y_{h-1}, a_{h-1})\right]$$

also the maximum span of the averaged advantage function and its discounted sum as follows;

$$\epsilon' = \max_{\tau\in\Upsilon} \overline{A}_{\pi_\theta,\pi_\theta}(y_{h+1}, y_h, h, y_{h-1}, a_{h-1})$$
$$\epsilon = \max_{\tau\in\Upsilon} \sum_{h}^{|\tau|} \gamma^h \overline{A}_{\pi_\theta,\pi_\theta}(y_{h+1}, y_h, h, y_{h-1}, a_{h-1})$$

Theorem 6.1 (Monotonic Improvement Guarantee)**.** *For two π_θ and $\pi_{\theta'}$, construct $L_{\pi_\theta}(\pi_{\theta'})$, then*

$$\eta(\pi_{\theta'}) \geq L_{\pi_\theta}(\pi_{\theta'}) - \epsilon TV\left(\tau \sim f(\cdot;\pi_{\theta'}), \tau \sim f(\tau;\pi_\theta)\right)$$
$$\geq L_{\pi_\theta}(\pi_{\theta'}) - \epsilon\sqrt{\frac{1}{2}\mathcal{D}_{KL}\left(\pi_{\theta'}, \pi_\theta\right)},$$
$$\eta(\pi_{\theta'}) \geq L_{\pi_\theta}(\pi_{\theta'}) - \epsilon'\sqrt{\mathcal{D}_\gamma\left(\pi_\theta, \pi_\theta\right)}.$$

Proof of Theorem 6.1 in the Appendix of Azizzadenesheli et al. (2018a).

The Theorem. 6.1 recommends optimizing $L_{\pi_\theta}(\pi_{\theta'})$ over $\pi_{\theta'}$ around the vicinity defined by $\mathcal{D}_{KL}$ or $\mathcal{D}_\gamma$ divergences Therefore, given the current policy π_θ we are interested in either of the following optimization:

$$\max_{\theta'} L_{\pi_\theta}(\pi_{\theta'}) - C\sqrt{\mathcal{D}_{KL}\left(\pi_\theta, \pi_{\theta'}\right)}$$
$$\max_{\theta'} L_{\pi_\theta}(\pi_{\theta'}) - C'\sqrt{\mathcal{D}_\gamma\left(\pi_\theta, \pi_{\theta'}\right))}$$

Where C and C' are the problem dependent constants. Similar to TRPO, using C and C' as they are might result in tiny changes in the policy. Therefore, for practical purposes, we view them as the knobs to restrict the trust region denoted by δ, δ' and turn these optimization problems to constraint optimization problems;

$$\max_{\theta'} L_{\pi_\theta}(\pi_{\theta'}) \quad s.t. \quad \mathcal{D}_{KL}\left(\pi_\theta, \pi_{\theta'}\right) \leq \delta$$
$$\max_{\theta'} L_{\pi_\theta}(\pi_{\theta'}) \quad s.t. \quad \mathcal{D}_{\gamma}\left(\pi_\theta, \pi_{\theta'}\right)) \leq \delta'$$

which results in Alg. 8. Taking into account the relationship between the KL divergence and Fisher information, we can also approximate these two optimization up to their second order Taylor expansion of the constraints;

$$\max_{\theta'} L_{\pi_\theta}(\pi_{\theta'}) \quad s.t. \quad \frac{1}{2}\| \left(\theta' - \theta\right)^\top F \left(\theta' - \theta\right) \|_2 \leq \delta$$
$$\max_{\theta'} L_{\pi_\theta}(\pi_{\theta'}) \quad s.t. \quad \frac{1}{2}\| \left(\theta' - \theta\right)^\top F_\gamma \left(\theta' - \theta\right) \|_2 \leq \delta'$$

These analyses provide insights into the design similar algorithm as TRPO and PPO for the general class of POMDPs.

6.5 Experiments

Extension to PPO: Usually, in high dimensional but low sample setting, constructing the trust region is hard due to high estimation errors. It is even harder especially when the region depends on the inverse of the estimated Fisher matrix or optimizing over the non-convex function of θ' with KL divergence constraint. Therefore, trusting the estimated trust region is questionable. While TRPO constructs the trust region in the parameter space, its final goal

is to keep the new policy close to the current policy, i.e., small $\mathcal{D}_{KL}(\pi_\theta, \pi_{\theta'})$ or $\mathcal{D}_\gamma(\pi_\theta, \pi_{\theta'})$. Proximal Policy Optimization (PPO) is instead proposed to impose the structure of the trust region directly onto the policy space. This method approximately translates the constraints developed in TRPO to the policy space. It penalized the gradients of the objective function when the policy starts to operate beyond the region of trust by setting the gradient to zero.

$$\mathbb{E}[\min\{\frac{\pi_{\theta'}(a|x)}{\pi_\theta(a|x)}\widetilde{A}_{\pi_\theta}(a,x), clip(\frac{\pi_{\theta'}(a|x)}{\pi_\theta(a|x)}; 1-\delta_L, 1+\delta_U)\widetilde{A}_{\pi_\theta}(a,x)\}]$$

We dropped the h dependency in the advantage function since this approach is for the infinite horizon MDPs. If the advantage function is positive, and the importance weight is above $1+\delta_U$ this objective function saturates. When the advantage function is negative, and the importance weight is below $1-\delta_L$ this objective function saturates again. In either case, when the objective function saturates, the gradient of this objective function is zero therefore further development in that direction is obstructed. This approach, despite its simplicity, approximates the trust region effectively and substantially reduce the computation cost of TRPO. **Note:** In the original PPO paper $\delta_U = \delta_L$.

Following the TRPO, the clipping trick ensures that the importance weight, derived from estimation of $\mathcal{D}_{KL}$ does not go beyond a certain limit $|\log \frac{\pi_{\theta'}(a|y)}{\pi_\theta(a|y)}| \leq \nu$, i.e.,

$$1-\delta_L := \exp(-\nu) \leq \frac{\pi_{\theta'}(a|y)}{\pi_\theta(a|y)} \leq 1+\delta_U := \exp(\nu) \tag{6.10}$$

As discussed in the Remark. 6.3.2 we propose a principled change in the clipping such that it matches Eq. 6.6 and conveys information about the length of episodes; $|\log \frac{\pi_\theta(a|y)}{\pi_{\theta'}(a|y)}| \leq \frac{\nu}{|\tau|}$; therefore for $\alpha := \exp(\nu)$

$$1-\delta_L := \alpha^{-1/|\tau|} \leq \frac{\pi_{\theta'}(a|y)}{\pi_{\theta_\theta(a|y)}} \leq 1+\delta_U := \alpha^{1/|\tau|} \tag{6.11}$$

This change ensures more restricted clipping for longer trajectories, while wider for shorter ones. Moreover, as it is suggested in theorem. 6.1, and the definition of $\mathcal{D}_\gamma(\pi_\theta, \pi_{\theta'})$ in Eq. 6.7, we propose a further extension in the clipping to conduct information about the discount factor. In order to satisfy $\mathcal{D}_\gamma\left(\pi_\theta, \pi_{\theta'}\right)) \leq \delta'$, for a sample at time step h of an episode we have $|\log \frac{\pi_{\theta'}(a|y)}{\pi_\theta(a|y)}| \leq \frac{\nu}{|\tau|\gamma^h}$. Therefore;

$$
\begin{aligned}
&1-\delta_L^h := \exp\left(-\frac{\nu}{|\tau|\gamma^h}\right) \leq \frac{\pi_{\theta'}(a|y)}{\pi_\theta(a|y)} \leq 1+\delta_U^h := \exp\left(\frac{\nu}{|\tau|\gamma^h}\right) \\
&\quad \rightarrow \alpha^{-1/|\tau|}\alpha^{-1/\gamma^h} \leq \frac{\pi_{\theta'}(a|y)}{\pi_\theta(a|y)} \leq \alpha^{1/|\tau|}\alpha^{1/\gamma^h}
\end{aligned} \tag{6.12}
$$

As it is interpreted, for deeper parts in the episode, we make the clipping softer and allow for larger changes in policy space. This means, we are more restricted at the beginning of trajectories compared to the end of trajectories. The choice of γ and α are critical here. In practical implementation of RL algorithm, as also theoretically suggested by Jiang et al. (2015); Lipton et al. (2016a) we usually choose discount factors smaller than the one for depicted in the problem. Therefore, the discount factor we use in practice is much smaller than the true one specially when we deploy function approximation. Therefore, instead of keeping γ^h in Eq. 6.12, since the true γ in practice is unknown and can be arbitrary close to 1, we substitute it with a maximum value;

$$
\begin{aligned}
1-\delta_L^h &:= \max\{\alpha^{-1/\left(|\tau|\gamma^h\right)}, 1-\beta\} \leq \frac{\pi_{\theta'}(a|y)}{\pi_\theta(a|y)} \\
&\leq 1+\delta_U^h := \min\{\alpha^{1/\left(|\tau|\gamma^h\right)}, 1+\beta\}
\end{aligned} \tag{6.13}
$$

The modification proposed in series of equations Eq. 6.10, Eq. 6.11, Eq. 6.12, and Eq6.13 provide insight in the use of trust regions in for both MDPs and POMDPs based PPO. The

PPO objective for any choice of δ_U^h and δ_L^h in MDPs is

$$\mathbb{E}\Big[\min\Big\{\frac{\pi_{\theta'}(a_h|x_h)}{\pi_\theta(a_h|x_h)}\widetilde{A}_{\pi_\theta}(a_h,x_h), clip(\frac{\pi_{\theta'}(a_h|x_h)}{\pi_\theta(a_h|x_h)}, 1-\delta_L^h, 1+\delta_U^h)\widetilde{A}_{\pi_\theta}(a_h,x_h)\Big\}\Big] \tag{6.14}$$

while for POMDPs we have

$$\begin{aligned}\mathbb{E}\Big[\min\Big\{&\frac{\pi_{\theta'}(a_h|y_h)}{\pi_\theta(a_h|x_h)}A_{\pi_\theta}(y_{h+1},a_h,y_h,y_{h-1},a_{h-1}),\\ &clip(\frac{\pi_{\theta'}(a_h|x_h)}{\pi_\theta(a_h|y_h)}, 1-\delta_L^h, 1+\delta_U^h)A_{\pi_\theta}(y_{h+1},a_h,y_h,y_{h-1},a_{h-1})\Big\}\Big]\end{aligned} \tag{6.15}$$

h is encoded in x_h. In order to make the existing MDP-based PPO suitable for POMDPs we just substitute $A_{\pi_\theta}(a_h,x_h)$ with $A_{\pi_\theta}(y_{h+1},a_h,y_h,y_{h-1},a_{h-1})$ in the corresponding line. Moreover, as we showed for TRPO, in the case of MDP model, Eq. 6.15 reduces to Eq. 6.14.

RoboSchool, a variant to MuJoCo: In the experimental study, we first started to analyze the behavior of the plain PPO agent but observe that the environment enforces a short termination which results in significantly short trajectories. We relaxed this hard threshold and analyzed PPO Appendix of Azizzadenesheli et al. (2018a). We deploy the analysis in Eq. 6.11 and Eq. 6.13, apply the suggested changes to the plain PPO and examine its performance in the variety of different parameters and environments Appendix of Azizzadenesheli et al. (2018a). As it is provided in the Appendix of Azizzadenesheli et al. (2018a), along with the mentioned experimental studies, we have done an extensive study on a variety of different settings to present a more detailed understanding of policy gradient methods. Throughout the experiments, we observe a similar behavior of the MDP based approach PPO and POMDP based approach GTRPO. This might be due to the simplicity of the environment as well as the close similarity of current state of environments are close to MDP. Along the course of the experimental study, we realized that the environment set-up and the deployed reward shaping require a critical and detailed modification to make the

test-bed suitable for further studies, Appendix of Azizzadenesheli et al. (2018a).

6.6 Conclusion

In this paper, we propose GTRPO, a trust region policy optimization method for the general class of POMDPs when the reward process given the current observation, current action, and successive observation is conditionally independent of the rest of variables. We develop a new advantage function for POMDPs which depends on three consecutive observation. The dependency on three consecutive observations also matches the claim in Azizzadenesheli et al. (2016c) which shows learning the model and minimizing the regret requires modeling three consecutive observations. GTRPO deploys this advantage function to perform the policy updates. We consider memoryless policies and show that each policy update derived by GTRPO is low variance and monotonically improves the expected return. Additionally, we show how to utilize the analyses in this work and extend the infinite horizon MDP based policy gradient methods, TRPO and PPO, to finite horizon MDPs as well as discounted reward setting. Finally, the same way that PPO extends TRPO and make it computationally more efficient, we extend GTRPO analyses and make it computationally more efficient. We implement this extension and empirical study its behavior along with PPO on Roboschool environments.

Chapter 7

Policy Gradient in Rich Observable MDPs

Reinforcement Learning in Rich-Observation MDPs using Spectral Methods

Reinforcement learning (RL) in Markov decision processes (MDPs) with large state spaces is a challenging problem. The performance of standard RL algorithms degrades drastically with the dimensionality of state space. However, in practice, these large MDPs typically incorporate a latent or hidden low-dimensional structure. In this paper, we study the setting of *rich-observation* Markov decision processes (ROMDP), where there are a small number of hidden states which possess an injective mapping to the observation states. In other words, every observation state is generated through a single hidden state, and this mapping is unknown a priori. We introduce a spectral decomposition method that consistently learns this mapping, and more importantly, achieves it with low regret. The estimated mapping is integrated into an optimistic RL algorithm (UCRL), which operates on the estimated hidden space. We derive finite-time regret bounds for our algorithm with a weak dependence on the dimensionality of the observed space. In fact, our algorithm asymptotically achieves

the same average regret as the oracle UCRL algorithm, which has the knowledge of the mapping from hidden to observed spaces. Thus, we derive an efficient spectral RL algorithm for ROMDPs.

7.1 Introduction

Reinforcement learning (RL) framework studies the problem of efficient agent-environment interaction, where the agent learns to maximize a given reward function in the long run (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998). At the beginning of the interaction, the agent is uncertain about the environment's dynamics and must *explore* different policies in order to gain information about it. Once the agent is fairly certain, the knowledge about the environment can be *exploited* to compute a good policy attaining a large cumulative reward. Designing algorithms that achieve an effective trade-off between exploration and exploitation is the primary goal of reinforcement learning. The trade-off is commonly measured in terms of *cumulative regret*, that is the difference between the rewards accumulated by the optimal policy (which requires exact knowledge of the environment) and those obtained by the learning algorithm.

In practice, we often deal with environments with large observation state spaces (e.g., robotics). In this case the regret of standard RL algorithms grows quickly with the size of the observation state space. (We use observation state and observation interchangeably.) Nonetheless, in many domains there is an underlying low dimensional latent space that summarizes the large observation space and its dynamics and rewards. For instance, in robot navigation, the high-dimensional visual and sensory input can be summarized into a 2D position map, but this map is typically unknown. This makes the problem challenging, since it is not immediately clear how to exploit the low-dimensional latent structure to achieve low regret.

Contributions. In this paper we focus on rich-observation Markov decision processes (ROMDP), where a small number of X hidden states are mapped to a large number of Y observations through an injective mapping, so that an observation can be generated by only one hidden state and hidden states can be viewed as clusters.

In this setting, we show that it is indeed possible to devise an algorithm that starting from observations can progressively cluster them in "smaller" states and eventually converge to the hidden MDP. We introduce SL-UCRL, where we integrate spectral decomposition methods into the upper-bound for RL algorithm (UCRL) (Jaksch et al., 2010a). The algorithm proceeds in epochs in which an estimated mapping between observations and hidden state is computed and an optimistic policy is computed on the MDP (called auxiliary MDP) constructed from the samples collected so far and the estimated mapping. The mapping is computed using spectral decomposition of the tensor associated to the observation process.

We prove that this method is guaranteed to correctly "cluster" observations together with high probability. As a result, the dimensionality of the auxiliary MDP decreases as more observations are clustered, thus making the algorithm more efficient computationally and more effective in finding good policies. Under transparent and realistic assumptions, we derive a regret bound showing that the per-step regret decreases over epochs, and we prove a worst-case bound on the number of steps (and corresponding regret) before the full mapping between states and observations is computed. The regret accumulated over this period is actually constant as the time to correct clustering does not increase with the number of steps N. As a result, SL-UCRL asymptotically matches the regret of learning directly on the latent MDP. We also notice that the improvement in the regret comes with an equivalent reduction in time and space complexity. In fact, as more observations are clustered, the space to store the auxiliary MDP decreases and the complexity of the extended value iteration step in UCRL decreases from $O(Y^3)$ down to $O(X^3)$.

Related work. The assumption of the existence of a latent space is often used to reduce the

learning complexity. For multi-armed bandits, Gheshlaghi azar et al. (2013) and Maillard and Mannor (2014) assume that a bandit problem is generated from an unknown (latent) finite set and show how the regret can be significantly reduced by learning this set. Gentile et al. (2014) consider the more general scenario of latent contextual bandits, where the contexts belong to a few underlying hidden classes. They show that a uniform exploration strategy over the contexts, combined with an online clustering algorithm achieve a regret scaling only with the number of hidden clusters. An extension to recommender systems is considered in Gopalan et al. (2016) where the contexts for the users and items are unknown a priori. Again, uniform exploration is used together with the spectral algorithm of Anandkumar et al. (2014) to learn the latent classes. Bartók et al. (Bartók et al., 2014) tackles a general case of partial monitoring games and provides minimax regret guarantee which is polynomial in certain dimensions of the problem.

The ROMDP model considered is a generalization of the latent contextual bandits, where actions influence the contexts (i.e., the states) and the objective is to maximize the long-term reward. ROMDPs have been studied in Krishnamurthy et al. (2016b) in the PAC-MDP setting and episodic deterministic environments using an algorithm searching the best Q-function in a given function space. This result is extended to the general class of contextual decision processes in Jiang et al. (2016). While the resulting algorithm is proved to achieve a PAC-complexity scaling with the number of hidden states/factors X, it suffers from high computations complexity.

Ortner (2013) proposes an algorithm integrating state aggregation with UCRL but, while the resulting algorithm may significantly reduce the computational complexity of UCRL, the analysis does not show any improvement in the regret.

Learning in ROMDPs can be also seen as a state-aggregation problem, where observations are aggregated to form a small latent MDP. While the literature on state-aggregation in RL is vast, most of the results have been derived for the batch setting (see e.g., Li et al. (2006)).

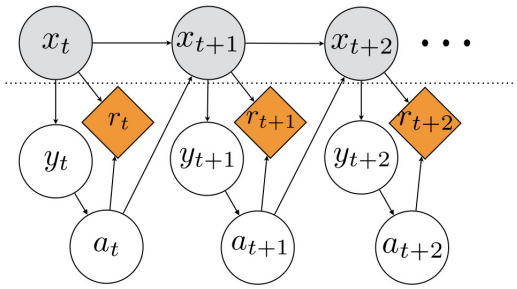


Figure 7.1: Graphical model of a ROMDP.

Finally, we notice that ROMDPs are a special class of partially observable MDPs (POMDP). Azizzadenesheli et al. (2016c) recently proposed an algorithm that leverages spectral methods to learn the hidden dynamic of POMDPs and derived a regret scaling as $\sqrt{Y}$ using fully stochastic policies (which are sub-optimal in ROMDPs). While the computation of the optimal memoryless policy relies on an optimization oracle, which in general is NP-hard Littman (1994); Vlassis et al. (2012); Porta et al. (2006); Azizzadenesheli et al. (2016a); Shani et al. (2013), computing the optimal policy in ROMDPs amounts to solving a standard MDP. Moreover, Guo et al. (2016) develops a PAC-MDP analysis for learning in episodic POMDPs and obtain a bound that depends on the size of the observations. The planning, in general, is computationally hard since it is a mapping from history to action.

7.2 Rich Observation MDPs

A rich-observation MDP (ROMDP) (Fig. 7.1) is a tuple $M = \langle \mathcal{X}, \mathcal{Y}, \mathcal{A}, R, f_T, f_O \rangle$, where $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{A}$ are the sets of hidden states, observations, and actions. We denote by X, Y, A their cardinality and we enumerate their elements by $i \in [X] = \{1..X\}$, $j \in [Y] = \{1..Y\}$, $l \in [A] = \{1..A\}$. We assume that the hidden states are fewer than the observations, i.e., $X \leq Y$. We consider rewards bounded in $[0, 1]$ that depend only on hidden states and actions with a reward matrix $R \in \mathbb{R}^{A \times X}$ such that $[R]_{i,l} = \mathbb{E}[r(x = i, a = l)]$. The dynamics of the MDP is defined on the hidden states as $T_{i',i,l} := f_T(i'|i, l) = \mathbb{P}(x' = i'|x = i, a = l)$, where

$T \in \mathbb{R}^{X \times X \times A}$ is the transition tensor. The observations are generated as $[O]_{j,i} = f_O(j|i) = \mathbb{P}(y=j|x=i)$, where the observation matrix $O \in \mathbb{R}^{Y \times X}$ has minimum *non-zero* entry $O_{\min}$. This model is a strict subset of POMDPs since each observation y can be generated by only one hidden state (see Fig. 7.2-*left*) and thus $\mathcal{X}$ can be seen as a non-overlapping clustering of the observations.

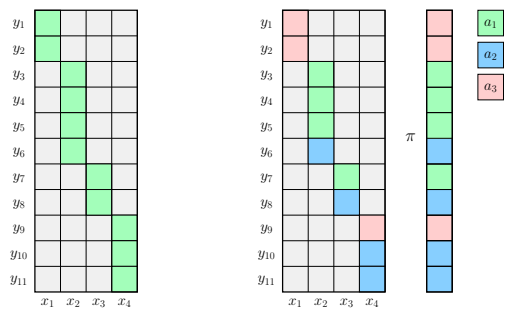


Figure 7.2: *(left)* Example of an observation matrix O. Since state and observation labeling is arbitrary, we arranged the non-zero values so as to display a diagonal structure. *(right)* Example of clustering that can be achieved by policy π (e.g., $\mathcal{X}_\pi^{(a_1)} = \{x_2, x_3\}$). Using each action we can recover *partial* clusterings corresponding to 7 auxiliary states $\mathcal{S} = \{s_1..s_7\}$ with clusters $\mathcal{Y}_{s_1} = \{y_1, y_2\}$, $\mathcal{Y}_{s_2} = \{y_3, y_4, y_5\}$, $\mathcal{Y}_{s_3} = \{y_6\}$, and $\mathcal{Y}_{s_8} = \{y_{10}, y_{11}\}$, while the remaining elements are the singletons y_6, y_7, y_8, and y_9. Clusters coming from different actions cannot be merged together because of different labeling of the hidden state, where, e.g., x_2 may be labeled differently depending on whether action a_1 or a_2 is used.

We denote by $\mathcal{Y}_x = \mathcal{Y}_i = \{y = j \in \mathcal{Y} : [O]_{j,i} > 0\}$ the set of observations in cluster x, while $x_y = x_j$ is the cluster observation $y = j$ belongs to.[1] This structure implies the existence of an *observable* MDP $M_\mathcal{Y} = \langle \mathcal{Y}, \mathcal{A}, R', f'_T \rangle$, where $R' = R$ as the reward of an observation-action pair (y, a) is the same as in the hidden state-action pair (x_y, a), and the dynamics can be obtained as $f'_T(j'|j, a) = \mathbb{P}(y' = j'|y = j, a = l) = \mathbb{P}(y' = j'|x' = x_{j'})\mathbb{P}(x' = x_{j'}|x = x_j, a = l) = [O]_{j',x_{j'}}[T]_{x_{j'},x_j,l}$. We measure the performance of an observation-based policy $\pi_\mathcal{Y} : \mathcal{Y} \to \mathcal{A}$ starting from a hidden state x by its asymptotic average reward $\rho(x; \pi_\mathcal{Y}) = \lim_{N\to\infty} \mathbb{E}\big[\sum_{t=1}^N r_t/N \big| x_0 = x, \pi_\mathcal{Y}\big]$. Given the mapping between the ROMDP to the hidden MDP, the optimal policy $\pi^*_\mathcal{Y}(y)$ is equal to the optimal hidden-state policy $\pi^*_\mathcal{X} : \mathcal{X} \to \mathcal{A}$ for all $y \in \mathcal{Y}_x$. The learning performance of an algorithm run over N steps is measured by the

[1] Throughout the paper we use the indices i, j, and l and the "symbolic" values x, y, and a interchangeably.

regret

$$R_N = N\rho^* - \big[\sum_{t=1}^{N} r_t\big], \ \textit{where } \rho^* = \rho(\pi^*_{\mathcal{X}}).$$

Finally we recall that the diameter of the observation MDP is defined as

$$D_{\mathcal{Y}} = \max_{y,y' \in \mathcal{Y}} \min_{\pi:\mathcal{Y}\to\mathcal{A}} \mathbb{E}\big[\tau_\pi(y,y')\big],$$

where $\tau_\pi(y,y')$ is the (random) number of steps from y to y' by following the observation-based policy π (similar for the diameter of the hidden MDP).

7.3 Learning ROMDP

In this section we introduce the spectral method used to learn the structure of the observation matrix O. In particular, we show that we do not need to estimate O exactly as the clusters $\{\mathcal{Y}_x\}_{x\in\mathcal{X}}$ can be recovered by identifying the non-zero entries of O. We need a first assumption on the ROMDP.

Assumption 7. *The Markov chain induced on the hidden MDP M by any policy $\pi_{\mathcal{Y}}$ is ergodic.*

Under this assumption for any policy π there is a stationary distribution over hidden states ω_π and a stationary distribution conditional on an action $\omega_\pi^{(l)}(i) = \mathbb{P}_\pi(x = i|a = l)$. Let $\mathcal{X}_\pi^{(l)} = \{i \in [X] : \omega_\pi^{(l)}(i) > 0\}$ be the hidden states where action l could be taken according to policy π. In other words, if $\mathcal{Y}_\pi^{(l)} = \{j \in [Y] : \pi(j) = l\}$ is the set of observations in which policy π takes action l, then $\mathcal{X}_\pi^{(l)}$ is the set of hidden states $\{x_y\}$ with $y \in \mathcal{Y}_\pi^{(l)}$ (see Fig. 7.2-*right*). We also define the set of all hidden states that can be reached starting from

states in $\mathcal{X}_\pi^{(l)}$ and taking action l, that is

$$\overline{\mathcal{X}}_\pi^{(l)} = \bigcup_{i \in \mathcal{X}_\pi^{(l)}} \Big\{ i' \in [X] : \mathbb{P}\big(x' = i' | x = i, a = l\big) > 0 \Big\}.$$

Similarly $\underline{\mathcal{X}}_\pi^{(l)}$ is the set of hidden states from which we can achieve the states $\mathcal{X}_\pi^{(l)}$ by policy π. We need the following assumption.

Assumption 8 (Full-Rank)**.** *Given any action l, the slice of transition tensor $[T]_{\cdot,\cdot,l}$ is full rank.*

Asm. 8 implies that for any action l the dynamics of M is "expansive", i.e., $|\mathcal{X}_\pi^{(l)}| \leq |\overline{\mathcal{X}}_\pi^{(l)}|$. In other words, the number of hidden states where policy π can take an action l (i.e., $\mathcal{X}_\pi^{(l)}$) is smaller than the number of states that can be reached when executing action l itself (i.e., $\overline{\mathcal{X}}_\pi^{(l)}$). These two assumptions ensure that the underlying Markov process is stochastic.

Multi-view model and exact recovery. We are now ready to introduce the multi-view model (Anandkumar et al., 2014) that allows us to reconstruct the clustering structure of the ROMDP Alg. 9. We consider the trajectory of observations and actions generated by an arbitrary policy π and we focus on three consecutive observations y_{t-1}, y_t, y_{t+1} at any step t. As customary in multi-view models, we *vectorize* the observations into three one-hot view vectors $\vec{v}_1$, $\vec{v}_2$, $\vec{v}_3$ in $\{0,1\}^Y$ such that $\vec{v}_1 = \vec{e}_j$ means that the observation in the first view is $j \in [Y]$ and where we remap time indices $t-1, t, t+1$ onto 1, 2, and 3. We notice that these views are indeed independent random variables when conditioning on the state x_2 (i.e., the hidden state at time t) and the action a_2 (i.e., the action at time t), thus defining a multi-view model for the hidden state process. Let $k_1 = |\underline{\mathcal{X}}_\pi^{(l)}|$, $k_2 = |\mathcal{X}_\pi^{(l)}|$ and $k_3 = |\overline{\mathcal{X}}_\pi^{(l)}|$, then we define the factor matrices $V_1^{(l)} \in \mathbb{R}^{Y \times k_1}, V_2^{(l)} \in \mathbb{R}^{Y \times k_2}, V_3^{(l)} \in \mathbb{R}^{Y \times k_3}$ as follows

$$[V_p^{(l)}]_{j,i} = \mathbb{P}(\vec{v}_p = \vec{e}_j | x_2 = i, a_2 = l),$$

where for $p{=}1,\ i \in \underline{\mathcal{X}}_\pi^{(l)}$, for $p{=}2,\ i \in \mathcal{X}_\pi^{(l)}$, and for $p{=}3,\ i \in \overline{\mathcal{X}}_\pi^{(l)}$.

Algorithm 9 Spectral learning algorithm.

Input: Trajectory $(y_1, a_1, \dots, y_N)$

For Action $l \in [A]$ **do**

Estimate second moments $\widehat{K}^{(l)}_{2,3}$, $\widehat{K}^{(l)}_{1,3}$, $\widehat{K}^{(l)}_{2,1}$, and $\widehat{K}^{(l)}_{3,1}$

Estimate the rank of matrix $\widehat{K}^{(l)}_{2,3}$ (see the Appendix of Azizzadenesheli et al. (2018a))

Compute symmetrized views $\widetilde{v}_{1,t}$ and $\widetilde{v}_{3,t}$, for $t = 2..N-2$

Compute second and third moments $\widehat{M}^{(l)}_2$ and $\widehat{M}^{(l)}_3$

Compute $\widehat{V}^{(l)}_2$ from the tensor decomposition of (an orthogonalized version of) $\widehat{M}^{(l)}_3$

return clusters

$$\widehat{\mathcal{Y}}^{(l)}_i = \{j \in [Y] : [\widetilde{V}^{(l)}_2]_{j,i} > 0\}$$

We are interested in estimating $V^{(l)}_2$ since it directly relates to the observation matrix as

$$[V^{(l)}_2]_{j,i} = \frac{\mathbb{P}(a_2 = l | y_2 = j)\mathbb{P}(y_2 = j | x_2 = i)}{\mathbb{P}(a_2 = l | x_2 = i)} = \frac{\mathbb{I}\{\pi(j) = l\} f_O(j|i)}{\mathbb{P}(a_2 = l | x_2 = i)}, \tag{7.1}$$

where $\mathbb{I}$ is the indicator function. As it can be noticed, $V^{(l)}_2$ borrows the same structure as the observation matrix O and since we want to recover only the clustering structure of M (i.e., $\{\mathcal{Y}_i\}_{i \in [X]}$), it is sufficient to compute the columns of $V^{(l)}_2$ up to any multiplicative constant. In fact, any non-zero entry of $V^{(l)}_2$ corresponds to a non-zero element in the original observation matrix (i.e., $[V^{(l)}_2]_{j,i} > 0 \Rightarrow [O]_{j,i} > 0$) and for any hidden state i, we can construct a cluster $\mathcal{Y}^{(l)}_i = \{j \in [Y] : [V^{(l)}_2]_{j,i} > 0\}$, which is accurate up to a re-labelling of the states. More formally, there exists a mapping function $\sigma^{(l)} : \mathcal{X} \to \mathcal{X}$ such that any pair of observations $j, j' \in \mathcal{Y}^{(l)}_i$ is such that $j, j' \in \mathcal{Y}_{\sigma(i)}$. Nonetheless, as illustrated in Fig. 7.2-*right*, the clustering may not be minimal. In fact, we have $[O]_{j,i} > 0 \not\Rightarrow [V^{(l)}_2]_{j,i} > 0$ since $[V^{(l)}_2]_{j,i}$ may be zero because of policy π, even if $[O]_{j,i} > 0$. Since the (unknown) mapping function $\sigma^{(l)}$ changes with actions, we are unable to correctly "align" the clusters and we may obtain more clusters than hidden states. We define $\mathcal{S}$ as the auxiliary state space obtained by the partial aggregation and we prove the following result.

Lemma 5. *Given a policy π, for any action l and any hidden state $i \in \mathcal{X}^{(l)}_\pi$, let $\mathcal{Y}^{(l)}_i$ be the observations that can be clustered together according to $V^{(l)}_2$ and $\mathcal{Y}^{\mathsf{c}} = \mathcal{Y} \setminus \bigcup_{i,l} \mathcal{Y}^{(l)}_i$ be the observations not clustered, then the auxiliary state space $\mathcal{S}$ contains all the clusters $\{\bigcup_{i,l} \mathcal{Y}^{(l)}_i\}$*

and the singletons in $\mathcal{Y}^c$ for a total number of elements $S = |\mathcal{S}| \leq AX$.

We now show how to recover the factor matrix $V_2^{(l)}$. We introduce mixed second and third order moments as $K_{p,q}^{(l)} = \mathbb{E}[\vec{v}_p \otimes \vec{v}_q], K_{p,q,r}^{(l)} = \mathbb{E}[\vec{v}_p \otimes \vec{v}_q \otimes \vec{v}_r]$ where p, q, r is any permutation of $\{1, 2, 3\}$. Exploiting the conditional independence of the views, the second moments can be written as

$$K_{p,q}^{(l)} = \sum_{i \in \mathcal{X}_\pi^l} \omega_\pi^{(l)}(i) [V_p^{(l)}]_{:,i} \otimes [V_q^{(l)}]_{:,i}$$

where $[V_p^{(l)}]_{:,i}$ is the i-th column of $V_p^{(l)}$. In general the second moment matrices are rank deficient, with rank $X_\pi^{(l)}$. We can construct a symmetric second moment by introducing the symmetrized views

$$\widetilde{v}_1 = K_{2,3}^{(l)} (K_{1,3}^{(l)})^\dagger \vec{v}_1, \qquad \widetilde{v}_3 = K_{2,1}^{(l)} (K_{3,1}^{(l)})^\dagger \vec{v}_3, \tag{7.2}$$

where $K^\dagger$ denotes the pseudoinverse. Then we can construct the second and third moments as

$$M_2^{(l)} = \mathbb{E}\big[\widetilde{v}_1 \otimes \widetilde{v}_3\big] = \sum_{i \in \mathcal{X}_\pi^{(l)}} \omega_\pi^{(l)}(i) [V_2^{(l)}]_{:,i} \otimes [V_2^{(l)}]_{:,i}. \tag{7.3}$$

$$M_3^{(l)} = \mathbb{E}\big[\widetilde{v}_1 \otimes \widetilde{v}_3 \otimes \vec{v}_2\big] = \sum_{i \in \mathcal{X}_\pi^l} \omega_\pi^{(l)}(i) [V_2^{(l)}]_{:,i} \otimes [V_2^{(l)}]_{:,i} \otimes [V_2^{(l)}]_{:,i}. \tag{7.4}$$

We can now employ the standard machinery of tensor decomposition methods to orthogonalize the tensor $M_3^{(l)}$ using $M_2^{(l)}$ and recover $V_2^{(l)}$ (refer to (Anandkumar et al., 2014) for further details) and a suitable clustering.

Lemma 6. *For any action $l \in [A]$, let $M_3^{(l)}$ be the third moment constructed on the symmetrized views as in Eq. 7.4, then we can orthogonalize it using the second moment $M_2^{(l)}$ and obtain a unique spectral decomposition from which we compute the exact factor matrix $[V_2^{(l)}]_{j,i}$. As a result, for any hidden state $i \in \mathcal{X}_\pi^{(l)}$ we define the cluster $\widetilde{\mathcal{Y}}_i^{(l)}$ as*

$$\widetilde{\mathcal{Y}}_i^{(l)} = \{j \in [Y] : [V_2^{(l)}]_{j,i} > 0\} \tag{7.5}$$

Algorithm 10 Spectral-Learning UCRL(SL-UCRL).

Initialize: $t = 1$, initial state x_1, $k = 1$, δ/N^6

While $t < N$ **do**

Run Alg. 9 on samples from epoch $k-1$ and obtain $\widehat{\mathcal{S}}$

Compute aux. space $\widehat{\mathcal{S}}^{(k)}$ by merging $\widehat{\mathcal{S}}$ and $\widehat{\mathcal{S}}^{(k-1)}$

Compute the estimate reward $r^{(k)}$ and dynamics $p^{(k)}$

Construct admissible AuxMDPs $\mathcal{M}^{(k)}$

Compute the optimistic policy

$$\widetilde{\pi}^{(k)} = \arg\max_{\pi} \max_{M \in \mathcal{M}^{(k)}} \rho(\pi; M) \tag{7.6}$$

Set $v^{(k)}(s,l) = 0$ for all actions $l \in \mathcal{A}, s \in \widehat{\mathcal{S}}^{(k)}$

While $\forall l, \forall s, v^{(k)}(s,l) < \max\{1, N^{(k)}(s,l)\}$ **do**

Execute $a_t = \widetilde{\pi}^{(k)}(s_t)$

Observe reward r_t and observation y_t

and there exists a mapping $\sigma^{(l)} : X \to X$ such that if $j, j' \in \widetilde{\mathcal{Y}}_i^{(l)}$ then $j, j' \in \mathcal{Y}_{\sigma^{(l)}(i)}$ (i.e., observations that are clustered together in $\widetilde{\mathcal{Y}}_i^{(l)}$ are clustered in the original ROMDP).

The computation complexity of Alg. 9 has been studied by Song et al. (2013) and is polynomial in the rank of third order moment.

Spectral learning. [2] While in practice we do not have the exact moments, we can only estimates them through samples. Let N be the length of the trajectory generated by policy π, then we can construct $N-2$ triples $\{y_{t-1}, y_t, y_{t+1}\}$ that can be used to construct the corresponding views $\vec{v}_{1,t}, \vec{v}_{2,t}, \vec{v}_{3,t}$ and to estimate second mixed moments as

$$\widehat{K}_{p,q}^{(l)} = \frac{1}{N(l)} \sum_{t=1}^{N(l)-1} \mathbb{I}(a_t = l)\ \vec{v}_{p,t} \otimes \vec{v}_{q,t},$$

with $p, q \in \{1,2,3\}$ and $N(l) = \sum_t^{N-1} \mathbb{I}(a_t = l)$. Furthermore, we require knowing $|\mathcal{X}_\pi^{(l)}|$, which is not known apriori. Under Asm. 7 and 8, for any action l, the rank of $K_{2,3}^{(l)}$ is indeed

[2] We report the spectral learning algorithm for the tensor decomposition but a very similar algorithm and guarantees can be derived for the matrix decomposition approach when the eigenvalues of $\widehat{M}_2^{(l)}$ for all actions and all posible policy have multiplicity 1. This further condition is not required when the tensor decomposition is deployed.

$|\mathcal{X}_\pi^{(l)}|$ and thus $\widehat{K}_{2,3}^{(l)}$ can be used to recover the rank. The actual way to calculate the efficient rank of $\widehat{K}_{2,3}^{(l)}$ is quite intricate and we represent the details in the Appendix of Azizzadenesheli et al. (2018a). From $\widehat{K}_{p,q}^{(l)}$ we can construct the symmetric views $\widetilde{v}_{1,t}$ and $\widetilde{v}_{3,t}$ as in Eq. 7.2 and compute the estimates of second and third moments as

$$\widehat{M}_2^{(l)} = \frac{1}{N(l)} \sum_{t=1}^{N-1} \mathbb{I}(a_t = l) \widetilde{v}_{1,t} \otimes \widetilde{v}_{3,t},$$
$$\widehat{M}_3^{(l)} = \frac{1}{N(l)} \sum_{t=1}^{N-1} \mathbb{I}(a_t = l) \widetilde{v}_{1,t} \otimes \widetilde{v}_{3,t} \otimes \vec{v}_{2,t}.$$

Following the same procedure as in the exact case, we are then able to recover estimates of the factor matrix $\widehat{V}_2^{(l)}$, which enjoys the following error guarantee.

Lemma 7. *Under Asm. 7 and 8, let $\widehat{V}_2^{(l)}$ be the empirical estimate of $V_2^{(l)}$ obtained using N samples generated by a policy π. There exists N_0 such that for any $N(l) > N_0$, $l \in \mathcal{A}$, $i \in \mathcal{X}_\pi^{(l)}$ w.p. $1-\delta$*

$$\|[V_2^{(l)}]_{\cdot,i} - [\widehat{V}_2^{(l)}]_{\cdot,i}\|_2 \leq C_2 \sqrt{\frac{\log(2Y^{3/2}/\delta)}{N(l)}} := \mathcal{B}_O^{(l)} \tag{7.7}$$

where C_2 is a problem-dependent constant independent from the number of observations Y.

While this estimate could be directly used to construct a clustering of observations, the noise in the empirical estimates might lead to $[\widehat{V}_2^{(l)}]_{j,i} > 0$ for any (j,i) pair, which prevents us from generating any meaningful clustering. On the other hand, we can use the guarantee in Lem. 7 to single-out the entries of $\widehat{V}_2^{(l)}$ that are non-zero w.h.p. We define the binary matrix $\widetilde{V}_2^{(l)} \in \{0,1\}^{Y \times X}$ as

$$[\widetilde{V}_2^{(l)}]_{j,i} = \begin{cases} 1 & \text{if } [\widehat{V}_2^{(l)}]_{j,i} \geq \mathcal{B}_O^{(l)} \\ 0 & \text{otherwise} \end{cases},$$

which relies on the fact that $[\widehat{V}_2^{(l)}]_{j,i} - \mathcal{B}_O^{(l)} > 0$ implies $[V_2^{(l)}]_{j,i} > 0$. At this point, for any l

and any $i \in \mathcal{X}_\pi^{(l)}$, we can generate the cluster

$$\widehat{\mathcal{Y}}_i^{(l)} = \{j \in [Y] : [\widetilde{V}_2^{(l)}]_{j,i} > 0\}, \tag{7.8}$$

which is guaranteed to aggregate observations correctly in high-probability. We denote be $\widehat{\mathcal{Y}}^{\mathsf{c}} = \mathcal{Y} \setminus \bigcup_{i,l} \widehat{\mathcal{Y}}_i^{(l)}$ the set of observations which are not clustered through this process. Then we define the auxiliary state space $\widehat{\mathcal{S}}$ obtained by enumerating all the elements of non-clustered observations together with clusters $\{\widehat{\mathcal{Y}}_i^{(l)}\}_{i,l}$, for which we have the following guarantee.

Corollary 1. *Let $\widehat{\mathcal{S}}$ be the auxiliary states composed of clusters $\{\widehat{\mathcal{Y}}_i^{(l)}\}$ and singletons in $\mathcal{Y}^{\mathsf{c}}$ obtained by clustering observations according to $\widetilde{V}_2^{(l)}$, then for any pair of observations j, j' clustered together in $\widehat{\mathcal{S}}$, there exists a hidden state i such that $j, j' \in \mathcal{Y}_i$. Finally, $\widehat{\mathcal{S}} \to \mathcal{S}$ as N tends to infinity.*

7.4 RL in ROMDP

We now describe the spectral learning UCRL (SL-UCRL) (Alg. 10) obtained by integrating the spectral method above with the UCRL strategy. The learning process is split into epochs of increasing length. At the beginning of epoch k, we use the trajectory $(s_1, a_1, .., s_{N^{(k-1)}})$ generated at previous epoch using auxiliary states $s \in \widehat{\mathcal{S}}^{(k)}$ to construct the auxiliary state space $\widehat{\mathcal{S}}$ using Alg. 9.[3] As discussed in the previous section, the limited number of samples and the specific policy executed at epoch $k-1$ may prevent from clustering many observations together, which means that despite $\widehat{\mathcal{S}}$ being a *correct* clustering (see Cor. 1), its size may still be large. While clusterings obtained at different epochs cannot be "aligned" because of different labelling, we can still effectively merge together any two clusterings $\widehat{\mathcal{S}}$ and $\widehat{\mathcal{S}}'$ generated by two different policies π and π'. We illustrate this procedure through Fig. 7.4.

[3]Since Alg. 9 receives as input a sequence of auxiliary states rather than observations as in the Appendix of Azizzadenesheli et al. (2018a), the spectral decomposition runs on a space of size $|\widehat{S}^{(k-1)}|$ instead of Y, thus reducing the computation complexity.

Observations y_3, y_4, and y_5 are clustered together in the auxiliary space generated by π, while y_5 and y_6 are clustered together using π'. While the labeling of the auxiliary states is arbitrary, observations preserve their labels across epochs and thus we can safely conclude that observations y_3, y_4, y_5, and y_6 belong to the same hidden state. Similarly, we can construct a new cluster with y_9, y_{10}, and y_{11}, which, in this case, returns the exact hidden space $\mathcal{X}$. Following this procedure we generate $\widehat{\mathcal{S}}^{(k)}$ as the clustering obtain by merging $\widehat{\mathcal{S}}$ and $\widehat{\mathcal{S}}^{(k-1)}$ (where $\widehat{\mathcal{S}}^1 = \mathcal{Y}$).

At this point we can directly estimate the reward and transition model of the auxiliary MDP constructed on $\widehat{\mathcal{S}}^{(k)}$ by using empirical estimators. For a sequence of clustering $\widehat{\mathcal{S}}^{(0)}, \ldots, \widehat{\mathcal{S}}^{(k)}$, since the clustering $\widehat{\mathcal{S}}^{(k)}$ is *monotonic* (i.e., observations clustered at epoch k stay clustered at any other epoch $k' > k$) any cluster $s^{(t^k)} \in \widehat{\mathcal{S}}^{(k)}$ can be represented as result of a *monotonically* aggregating observations as a increasing series of $s^1 \subseteq s^2 \subseteq \ldots \subseteq s^{(t^k-1)}$ (not unique, and random. As it is has been shown in Fig. 7.3 any branching can be considered as one of these series. Let's choose one of them. Here, s^t is a cluster at time point $t(\leq t_k)$ which evolves to

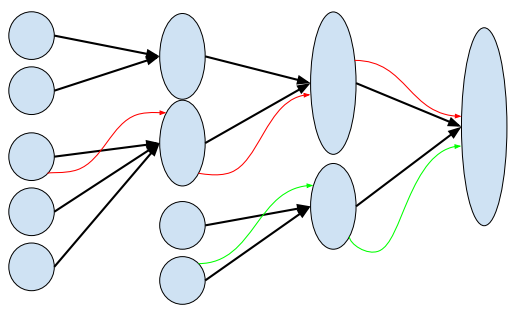


Figure 7.3: Monotonic evolution of clusters, each layer is the beginning of an epoch. The green and red paths are two examples for two different cluster aggregation.

the cluster $s^{(t_k)}$. For a clustering sequence $s^1 \subseteq s^2 \subseteq \ldots \subseteq s^{(t^k-1)}$, evolving to $s^{(t^k)}$, define $N^{(k)}(s, a)$, the number of samples in interest is:

$$N^{(k)}(s^{(t^k)}, a) = \sum_{t}^{t^{(k)}} \mathbb{1}(y_t \in s^t)\mathbb{1}(a_t = a)$$

with an abuse of notation, we write $y \in s^t$ to denote that the observation y has been clustered into an auxiliary state s^t at time step t. For any observation y, we use all the samples of y

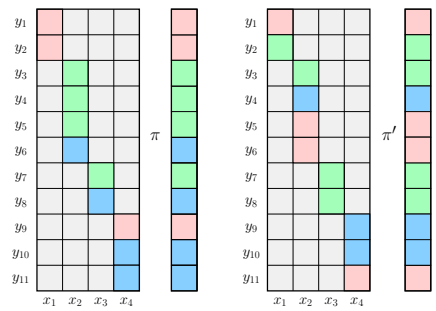


Figure 7.4: Examples of clusterings obtained from two policies that can be effectively merged.

to decide whether to merge this observation to any cluster. When we merge this observation to a cluster, we do not use the past sample of y for the empirical estimates of reward and transition. For example, Fig. 7.4, we cluster together $\{y_3, y_4, y_5\}$. At the beginning of each epoch, we use all the samples to decide whether y_6 belongs to this cluster. For an epoch, when we decide that y_6 belongs to this cluster, we do not use the samples of y_6 up to this epoch to estimate the reward and transition estimates of cluster $\{y_3, y_4, y_5, y_6\}$. Therefore, to estimate the empirical mean of reward and the transition kernel, we have

$$\widehat{r}^{(k)}(s^{(t^k)}, a) = \sum_{t}^{t^{(k)}} r_t \mathbb{1}(y_t \in s^t)\mathbb{1}(a_t = a)/N^{(k)}(s, a)$$

and for transitions, let's define the following count

$$N^{(k)}(s^{(t^k)}, a, s') = \sum_{t}^{t^{(k)}} \mathbb{1}(y_{t+1} \in s')\mathbb{1}(y_t \in s^t)\mathbb{1}(a_t = a)$$

therefore

$$\widehat{p}^{(k)}(s'|s^{(t^k)}, a) = N^{(k)}(s^{(t^k)}, a, s')/N^{(k)}(s^{(t^k)}, a)$$

then we return the estimates.[4] For further use, we define the per-epoch samples of interest for $s \in \widehat{\mathcal{S}}^{(k)}$ as $\nu^{(k)}(s^{(t^k)}, a) := \sum_{y \in \mathcal{Y}} \sum_{t^{(k-1)}}^{t^{(k)}} z_{t,t^{(k)}}(y)$ The corresponding confidence intervals

[4]Since the clustering $\widehat{\mathcal{S}}^{(k)}$ is *monotonic*, $\widehat{r}^{(k)}$ and $\widehat{p}^{(k)}$ can be computed incrementally without storing the statistics $N^{(k)}(y, a, y')$, $N^{(k)}(y, a)$, and $R^{(k)}(y, a)$ at observation level, thus significantly reducing the space complexity of the algorithm.

are such that for any $s \in \widehat{\mathcal{S}}^{(k)}$ and $a \in \mathcal{A}$

$$\|p(\cdot|s,a) - \widehat{p}^{(k)}(\cdot|s,a)\|_1 \leq d_p(s,a) = \sqrt{\frac{28 S^{(k)} \log(\frac{2AN^{(k)}}{\delta})}{N^{(k)}(s,a)}},$$

$$|\bar{r}(s,a) - \widehat{r}^{(k)}(s,a)| \leq d_r(s,a) = \sqrt{\frac{28 \log(\frac{2YAN^{(k)}}{\delta})}{N^{(k)}(s,a)}},$$

hold w.p. $1-\delta$, where $p(\cdot|s,a)$ and $\bar{r}$ are the transition probabilities and reward of the auxiliary MDP $M_{\widehat{\mathcal{S}}^{(k)}}$ Appendix of Azizzadenesheli et al. (2018a). Given the estimates and the confidence intervals, we construct a set of plausible auxiliary MDPs, $\mathcal{M}^{(k)}$, where the reward means and transition probabilities satisfy the confidence intervals.

At this point we can simply apply the same steps as in standard UCRL, where an optimistic auxiliary MDP $\widetilde{M}^{(k)}$ is constructed using the confidence intervals above and extended value iteration (EVI) (Jaksch et al., 2010a). The resulting optimal optimistic policy $\widetilde{\pi}^{(k)}$ is then executed until the number of samples at least for one pair of auxiliary state and action is doubled.

EVI has a per-iteration complexity which scales as $\mathcal{O}((\widehat{S}^{(k)})^2 A)$ thus gradually reducing the complexity of UCRL on the observation space (i.e., $\mathcal{O}((Y)^2 A)$) as soon as observations are clustered together. When the whole clustering is learnt, the computational complexity of EVI tends to $\mathcal{O}((X)^2 A)$. Moreover, since we aggregate the samples of the elements in clusters, therefore more accurate estimates, the number of times we call EVI algorithm goes from $\mathcal{O}(Y \log(N))$ to $\mathcal{O}(X \log(N))$.

Theorem 7.1. *Consider a ROMDP $M = \langle \mathcal{X}, \mathcal{Y}, \mathcal{A}, R, f_T, f_O \rangle$ with diameter $D_{\mathcal{X}}$. If* SL-UCRL *is run over N time steps, under Asm. 7 and 8, with probability $1-\delta$ it suffers the total regret of*

$$Reg_N \leq \sum_{k=1}^{K} \Big(D_{\widehat{S}^{(k)}} \sqrt{\widehat{S}^{(k)} \log\Big(\frac{N^{(k)}}{\delta}\Big)} \sum_{s \in \widehat{\mathcal{S}}^{(k)}, a} \frac{\nu^{(k)}(s,a)}{\sqrt{N^{(k)}(s,a)}} \Big),$$

where $(\mathcal{S}^{(k)})$ is the sequence of auxiliary state spaces generated over K epochs.

Remark. This bound shows that the per-step regret decreases over epochs. First we notice that only the regret over the first few (and short) epochs actually depends on the number of observations Y and the diameter $D_{\mathcal{Y}}$. As soon as a few observations start being clustered into auxiliary states, the regret depends on the number of auxiliary states $\widehat{S}^{(k)}$ and the diameter $D_{\mathcal{S}^{(k)}}$. Since $\widehat{S}^{(k)}$ decreases every time an observation is added to a cluster and $D_{\mathcal{S}^{(k)}}$ is monotonically decreasing with of $\widehat{\mathcal{S}}^{(k)}$, the per-step regret significantly decreases with epochs.[5] Cor. 1 indeed guarantees that the number of auxiliary states in $\widehat{\mathcal{S}}$ reduces down to $|\mathcal{S}|$ (XA in the worst case) as epochs get longer. Furthermore we recall that even if the clustering $\widehat{\mathcal{S}}$ returned by the spectral method is not minimal, merging clusters across epochs may rapidly result in very compact representations even after a few epochs.

Minimal clustering. While Thm. 7.1 shows that the performance of SL-UCRL improves over epochs, it does not relate it to the (ideal) performance that could be achieved when the hidden space had been known. Unfortunately, even if the number of clusters in $\widehat{\mathcal{S}}^{(k)}$ is nearly-minimal, the MDP constructed on the auxiliary state space may have a large diameter. In fact, it is enough that an observation j with very low probability $O_{j,i}$ is not clustered (it is a singleton in $\mathcal{S}^{(k)}$) to have a diameter that scales as $1/O_{\min}$ (although its *actual* impact on the regret may be negligible, for instance when j is not visited by the current policy). , in general the advantage obtained by clustering reduces the dependency on the number of states from Y to XA but it may not be effective in reducing the dependency on the diameter from $D_{\mathcal{Y}}$ to $D_{\mathcal{X}}$.

In order to provide a minimal clustrting, we integrate Alg. 10 with a clustering technique similar to the one used in Gentile et al. (2014) and Ortner (2013). At any epoch k, we proceed by merging together all the auxiliary states in $\widehat{\mathcal{S}}^{(k)}$ whose reward and transition confidence intervals overlap (i.e., s and s' are merged if the confidence interval $[\widehat{r}(s,a) \pm d_r(s,a)]$ overlaps

[5] We refer to the per-step regret since an epoch may be longer, thus making the cumulative epoch regret larger.

with $[\widehat{r}(s',a) \pm d_r(s',a)]$ and $[\widehat{p}(\cdot|s,a) \pm d_p(s,a)]$[6] overlaps with $[\widehat{p}(\cdot|s',a) \pm d_p(s',a)]$. If the number of new clusters is equal to X, then we claim we learned the true clustering, if it is less than X we ignore this temporary clustering and proceed to the next epoch. It is worth noting that this procedure requires the knowledge of X, while the spectral method, by its own, does not. While an explicit rate of clustering is very difficult to determine (the merging process depends on the spectral method, whose result depends on the policy, which in turn is determined according to the clustering at previous epochs), we derive worst-case bounds on the number of steps needed to start clustering at least one observation (i.e., steps before avoiding the dependency on Y and $D_{\mathcal{Y}}$) and before the exact clustering is recovered.

Corollary 2. *Let $\tau_M = \max_{x,\pi} \mathbb{E}_\pi[\tau_\pi(x,x)]$ the maximum expected returning time in MDP M (bounded due to ergodicity) and*

$$\begin{aligned}\overline{N}_{first} &= \frac{AY\tau_M}{O_{\min}} \frac{C_2 \log(1/\delta)}{\max_{i,j} f_O(y=j|x=i)^2};\\ \overline{N}_{last} &= \frac{AY\tau_M}{O_{\min}^3} C_2 \log(1/\delta).\end{aligned} \tag{7.9}$$

After $\overline{N}_{first}$ steps at least two observations are clustered and after $\overline{N}_{last}$ steps all observations are clustered (but not necessarily in the minimum hidden space configuration) with probability $1-\delta$. This implies that after $\overline{N}_{last}$ steps $|\widehat{S}^{(k)}| \leq XA$. Furthermore, let $\gamma_r = \min_{x,x',a} |r(x,a) - r(x',a)|$ and $\gamma_p = \min_{x,x',a} \|p(\cdot|x,a) - p(\cdot|x',a)\|_1$ be the smallest gaps between rewards and transition probabilities and let $\gamma = \max\{\gamma_r, \gamma_p\}$ the maximum between the two. In the worst case, using the additional clustering step together with SL-UCRL *guarantees that after $\overline{N}_{\mathcal{X}}$*

$$\min\Big\{\frac{AY^2\tau_M}{\gamma^2}\log(1/\delta), \max\Big\{\frac{AS^2\tau_M}{\gamma^2}\log(1/\delta), \overline{N}_{last}\Big\}\Big\}$$

samples the hidden state $\mathcal{X}$ is correctly reconstructed (i.e., $\widehat{\mathcal{S}}^{(k)} = \mathcal{X}$), therefore

$$\begin{aligned}Reg_N \leq\ & 34 D_{\mathcal{X}} X \sqrt{A(N-\overline{N}_{\mathcal{X}})\log(N/\delta)}\mathbb{I}(N \geq \overline{N}_{\mathcal{X}})\\ &+ \min\{\overline{N}_{\mathcal{X}}, 34 D_{\mathcal{Y}} Y \sqrt{A(\overline{N}_{\mathcal{X}})\log(N_{\mathcal{X}}/\delta)}\}\end{aligned}$$

[6] Deviation $d_p(s,a)$ on a $\widehat{S}$ dimensional simplex

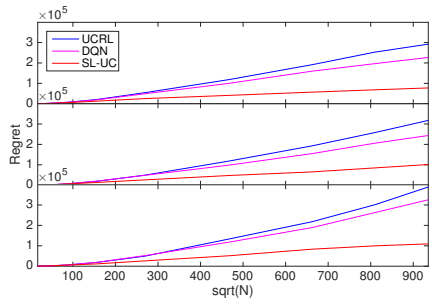


Figure 7.5: Regret comparison for ROMDPs with $X = 5, A = 4$ and from top to bottom $Y = 10, 20, 30$.

We first notice that this analysis is constructed over a series of worst-case steps (see proof in the Appendix of Azizzadenesheli et al. (2018a)). Nonetheless, it first shows that the number of observations Y does impact the regret only over the first $\overline{N}_{\text{first}}$ steps, after which $\widehat{S}^{(k)}$ is already smaller than $\mathcal{Y}$. Furthermore, after at most $\overline{N}_{\text{last}}$ the auxiliary space has size at most XA (while the diameter may still be as large as $D_{\mathcal{Y}}$). Finally, after $\overline{N}_{\mathcal{X}}$ steps $\widehat{S}^{(k)}$ reduces to $\mathcal{X}$ and the performance of SL-UCRL tends to the same performance of UCRL in the hidden MDP.

7.5 Experiments

We validate our theoretical results by comparing the performance of SL-UCRL, UCRL2 (model based) and DQN (model free, function approximation) Mnih et al. (2015), two well known RL algorithms. The goal of this experiment is to evaluate the dependency of regret to dimensionality of observation space. Generally, DQN is considered as model free RL method which extend the notion of Bellman residual (Antos et al., 2008) to deep RL. For DQN, we implement a three layers feed forward network (with no CNN block), equipped with RMSprop and replay buffer. We tune the hyper parameters of the network and report the best performance achieved by network of size $30 \times 30 \times 30$.

We consider three randomly generated ROMDPs (Dirichlet transition and Uniform reward with different bias) with $X = 5$, $A = 4$ and observation spaces of sizes $Y = 10, 20, 30$. Fig. 7.5 reports the regret on a $\sqrt{N}$ scale where regret of UCRL and DQN grows much faster than SL-UCRL's. While all regrets tend to be linear (i.e., growing as $\sqrt{N}$), we observe that the regret slope of UCRL and DQN are negatively affected by the increasing number of observations, while the regret slope of SL-UCRL stays almost constant, confirming that the hidden space $\mathcal{X}$ is learned rapidly. These experiments are the first step towards more practical applications. Additional experiments in the Appendix of Azizzadenesheli et al. (2018a).

7.6 Conclusion

We introduced SL-UCRL, a novel RL algorithm to learn in ROMDPs combining a spectral method for recovering the clustering structure of the problem and UCRL to effectively trade off exploration and exploitation. We proved theoretical guarantees showing that SL-UCRL progressively refines the clustering so that its regret tends to the regret that could be achieved when the hidden structure is known in advance (in higher order term). Despite this result almost matching the regret obtained by running UCRL directly on the latent MDP, the regret analysis requires ergodicity of the MDP. One of the main open questions is whether the spectral clustering method could still provide "useful" clusterings when the state space is not fully visited (i.e., in case of non-ergodic MDP), so that observations are properly clustered where it is actually needed to learn the optimal policy the Appendix of Azizzadenesheli et al. (2018a).

At the beginning of the learning, the algorithm deals with larger MDPs and gradually, while learning bigger cluster, it starts to deal with smaller MDPs. Reducing the state space of MDPs means lower cost in computing optimistic policy, having fewer number of epochs, and

suffer from lower computation cost. Finally, this work opens several interesting directions to extend the results for variety of state aggregation topologies (Li et al., 2006).

Bibliography

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. (2011). Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320.

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. (2011). Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems 24 - NIPS*, pages 2312–2320.

Abbasi-Yadkori, Y., Pal, D., and Szepesvari, C. (2012). Online-to-confidence-set conversions and application to sparse stochastic bandits. In *Artificial Intelligence and Statistics*, pages 1–9.

Abbasi-Yadkori, Y. and Szepesvári, C. (2011). Regret bounds for the adaptive control of linear quadratic systems. In *COLT 2011 - The 24th Annual Conference on Learning Theory, June 9-11, 2011, Budapest, Hungary*.

Abbasi-Yadkori, Y. and Szepesvári, C. (2011). Regret bounds for the adaptive control of linear quadratic systems. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 1–26.

Abbasi-Yadkori, Y. and Szepesvári, C. (2015). Bayesian optimal control of smoothly parameterized systems. In *UAI*, pages 1–11.

Abeille, M. and Lazaric, A. (2017). Linear thompson sampling revisited. In *AISTATS 2017-20th International Conference on Artificial Intelligence and Statistics*.

Abel, D., Agarwal, A., Diaz, F., Krishnamurthy, A., and Schapire, R. E. (2016). Exploratory gradient boosting for reinforcement learning in complex domains. *arXiv*.

Agrawal, S. and Goyal, N. (2012). Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*.

Agrawal, S. and Goyal, N. (2013). Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, pages 127–135.

Akhiezer, N. I. and Glazman, I. M. (2013). *Theory of linear operators in Hilbert space*. Courier Corporation.

Aleksandrov, V. M., Sysoyev, V. I., and Shemeneva, V. V. (1968). Stochastic optimaization. *Engineering Cybernetics*, 5(11-16):229–256.

Amari, S.-i. (2016). *Information geometry and its applications*. Springer.

Anandkumar, A., Ge, R., Hsu, D., Kakade, S. M., and Telgarsky, M. (2014). Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832.

Anandkumar, A., Hsu, D., and Kakade, S. M. (2012). A method of moments for mixture models and hidden markov models. *arXiv preprint arXiv:1203.0683*.

Anderson, T. W. et al. (1963). Asymptotic theory for principal component analysis. *Annals of Mathematical Statistics*, 34(1):122–148.

Antos, A., Szepesvári, C., and Munos, R. (2008). Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*.

Asmuth, J., Li, L., Littman, M. L., Nouri, A., and Wingate, D. (2009). A bayesian sampling approach to exploration in reinforcement learning. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*.

Atrash, A. and Pineau, J. (2006). Efficient planning and tracking in pomdps with large observation spaces. In *AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*.

Auer, P. (2003). Using confidence bounds for exploitation-exploration trade-offs. *The Journal of Machine Learning Research*, 3:397–422.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256.

Auer, P., Jaksch, T., and Ortner, R. (2009). Near-optimal regret bounds for reinforcement learning. In *Advances in neural information processing systems*, pages 89–96.

Azizzadenesheli, K. (2019). Maybe a few considerations in reinforcement learning research?

Azizzadenesheli, K. and Anandkumar, A. (2018). Efficient exploration through bayesian deep q-networks. *arXiv preprint arXiv:1802.04412*.

Azizzadenesheli, K., Bera, M. K., and Anandkumar, A. (2018a). Trust region policy optimization of pomdps. *arXiv preprint arXiv:1810.07900*.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. (2016a). Open problem: Approximate planning of pomdps in the class of memoryless policies. In *Conference on Learning Theory*, pages 1639–1642.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. (2016b). Reinforcement learning in rich-observation mdps using spectral methods. *arXiv preprint arXiv:1611.03907*.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. (2016c). Reinforcement learning of pomdps using spectral methods. *arXiv preprint arXiv:1602.07764*.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. (2017). Experimental results: Reinforcement learning of pomdps using spectral methods. *arXiv preprint arXiv:1705.02553*.

Azizzadenesheli, K., Liu, A., Yang, F., and Anandkumar, A. (2019). Regularized learning for domain adaptation under label shifts. *arXiv preprint arXiv:1903.09734*.

Azizzadenesheli, K., Yang, B., Liu, W., Brunskill, E., Lipton, Z., and Anandkumar, A. (2018b). Surprising negative results for generative adversarial tree search. *PGMRL workshop at ICML*.

Bagnell, J. A., Kakade, S. M., Schneider, J. G., and Ng, A. Y. (2004). Policy search by dynamic programming. In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems 16*, pages 831–838. MIT Press.

Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Elsevier*.

Bartlett, P. L. and Tewari, A. (2009). REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence*.

Barto, A., Sutton, R., and Anderson, C. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-13(5):834–846.

Bartók, G., Foster, D. P., Pál, D., Rakhlin, A., and Szepesvári, C. (2014). Partial monitoring—classification, regret bounds, and algorithms. *Mathematics of Operations Research*.

Baxter, J. and Bartlett, P. L. (2001a). Infinite-horizon policy-gradient estimation. *J. Artif. Int. Res.*, 15(1):319–350.

Baxter, J. and Bartlett, P. L. (2001b). Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.(JAIR)*.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. (2018a). Compression by the signs: distributed learning is a two-way street.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. (2018b). signsgd: compressed optimisation for non-convex problems. *arXiv preprint arXiv:1802.04434*.

Bernstein, J., Zhao, J., Azizzadenesheli, K., and Anandkumar, A. (2018c). signsgd with majority vote is communication efficient and byzantine fault tolerant. *arXiv preprint arXiv:1810.05291*.

Bertsekas, D. and Tsitsiklis, J. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Boots, B., Siddiqi, S. M., and Gordon, G. J. (2011). Closing the learning-planning loop with predictive state representations. *The International Journal of Robotics Research*, 30(7):954–966.

Bora, A., Jalal, A., Price, E., and Dimakis, A. G. (2017). Compressed sensing using generative models. *arXiv preprint arXiv:1703.03208*.

Boyan, J. and Moore, A. W. (1995). Generalization in reinforcement learning: Safely approximating the value function. In *NIPS*.

Brafman, R. I. and Tennenholtz, M. (2003). R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning Research*, 3:213–231.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym.

Candes, E. J. and Tao, T. (2006). Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE transactions on information theory*, 52(12):5406–5425.

Carpentier, A. and Munos, R. (2012). Bandit theory meets compressed sensing for high dimensional stochastic linear bandit. In *Artificial Intelligence and Statistics*, pages 190–198.

Chapelle, O. and Li, L. (2011). An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257.

Chaudhari, P. and Soatto, S. (2018). Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–10. IEEE.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv*.

Chen, W., Wang, Y., and Yuan, Y. (2013). Combinatorial multi-armed bandit: General framework and applications. In Dasgupta, S. and Mcallester, D., editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 151–159.

Chentanez, N., Barto, A. G., and Singh, S. P. (2004). Intrinsically motivated reinforcement learning. In *NIPS*.

Chow, Y., Tamar, A., Mannor, S., and Pavone, M. (2015). Risk-sensitive and robust decision-making: A CVaR optimization approach. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1522–1530.

Dani, V., Hayes, T. P., and Kakade, S. M. (2008). Stochastic linear optimization under bandit feedback. *COLT*.

Davis, C. and Kahan, W. M. (1970). The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1):1–46.

Dearden, R., Friedman, N., and Russell, S. (1998). Bayesian q-learning. In *AAAI/IAAI*, pages 761–768.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A., and Anandkumar, A. (2018). Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*.

Engel, Y., Mannor, S., and Meir, R. (2003). Bayes meets bellman: The gaussian process approach to temporal difference learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*.

Fatemi, M., El Asri, L., Schulz, H., He, J., and Suleman, K. (2016). Policy networks with two-stage training for dialogue systems. In *SIGDIAL*.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., et al. (2017). Noisy networks for exploration. *arXiv*.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*.

Garcıa, J. and Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *JMLR*.

Gentile, C., Li, S., and Zappella, G. (2014). Online clustering of bandits. In *International Conference on Machine Learning*, pages 757–765.

Ghavamzadeh, M., Mannor, S., Pineau, J., Tamar, A., et al. (2015). Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*.

Gheshlaghi-Azar, M., Lazaric, A., and Brunskill, E. (2013). Regret bounds for reinforcement learning with policy advice. In *Proceedings of the European Conference on Machine Learning (ECML'13)*.

Gheshlaghi azar, M., Lazaric, A., and Brunskill, E. (2013). Sequential transfer in multi-armed bandit with finite set of models. In Burges, C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 26*, pages 2220–2228. Curran Associates, Inc.

Gheshlaghi-Azar, M., Lazaric, A., and Brunskill, E. (2014). Resource-efficient stochastic optimization of a locally smooth function under correlated bandit feedback. In *Proceedings of the Thirty-First International Conference on Machine Learning (ICML'14)*.

Gopalan, A., Maillard, O.-A., and Zaki, M. (2016). Low-rank bandits with latent mixtures. *arXiv preprint arXiv:1609.01508*.

Gordon, G. J. (1996). Chattering in SARSA(λ). Technical report, CMU.

Guo, Z. D., Doroudi, S., and Brunskill, E. (2016). A PAC rl algorithm for episodic POMDPs. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 510–518.

Hanneke, S. (2016). The optimal sample complexity of PAC learning. *Journal of Machine Learning Research*.

Hans, A., Schneegaß, D., Schäfer, A. M., and Udluft, S. (2008). Safe exploration for reinforcement learning. In *ESANN*.

Hauskrecht, M. and Fraser, H. (2000). Planning treatment of ischemic heart disease with partially observable markov decision processes. *Artificial Intelligence in Medicine*, 18(3):221 – 244.

Heger, M. (1994). Consideration of risk in reinforcement learning. In *Machine Learning*.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. (2017). Deep reinforcement learning that matters. *arXiv*.

Hsu, D., Kakade, S. M., and Zhang, T. (2012). Random design analysis of ridge regression. In *Conference on Learning Theory*, pages 9–1.

Hsu, D. J., Kontorovich, A., and Szepesvári, C. (2015). Mixing time estimation in reversible markov chains from a single sample path. In *Advances in neural information processing systems*, pages 1459–1467.

Hyvärinen, A. and Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430.

Jaakkola, T., Singh, S. P., and Jordan, M. I. (1995). Reinforcement learning algorithm for partially observable markov decision problems. In *Advances in Neural Information Processing Systems 7*, pages 345–352. MIT Press.

Jain, P., Jin, C., Kakade, S. M., Netrapalli, P., and Sidford, A. (2016). Streaming pca: Matching matrix bernstein and near-optimal finite sample guarantees for oja's algorithm. In *Conference on Learning Theory*, pages 1147–1164.

Jaksch, T., Ortner, R., and Auer, P. (2010a). Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600.

Jaksch, T., Ortner, R., and Auer, P. (2010b). Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11:1563–1600.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. (2016). Contextual decision processes with low bellman rank are pac-learnable. *arXiv preprint arXiv:1610.09512*.

Jiang, N., Kulesza, A., Singh, S., and Lewis, R. (2015). The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1181–1189.

Kakade, S., Kearns, M. J., and Langford, J. (2003). Exploration in metric state spaces. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 306–312.

Kakade, S. and Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *ICML*, volume 2, pages 267–274.

Kakade, S. M. (2002). A natural policy gradient. In *Advances in neural information processing systems*, pages 1531–1538.

Kearns, M. and Singh, S. (2002). Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232.

Kleinberg, R., Niculescu-Mizil, A., and Sharma, Y. (2010). Regret bounds for sleeping experts and bandits. *Machine learning*, 80(2-3):245–272.

Kocsis, L. and Szepesvári, C. (2006). Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pages 282–293. Springer.

Kontorovich, A., Nadler, B., and Weiss, R. (2013). On learning parametric-output hmms. *arXiv preprint arXiv:1302.6009*.

Kontorovich, A., Weiss, R., et al. (2014). Uniform chernoff and dvoretzky-kiefer-wolfowitz-type inequalities for markov chains and related processes. *Journal of Applied Probability*, 51(4):1100–1113.

Kontorovich, L. A., Ramanan, K., et al. (2008). Concentration inequalities for dependent random variables via the martingale method. *The Annals of Probability*, 36(6):2126–2158.

Krishnamurthy, A., Agarwal, A., and Langford, J. (2016a). Contextual-mdps for pac-reinforcement learning with rich observations. *arXiv preprint arXiv:1602.02722v1*.

Krishnamurthy, A., Agarwal, A., and Langford, J. (2016b). PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, pages 1840–1848.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical report, Citeseer.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Lagoudakis, M. G. and Parr, R. (2003). Least-squares policy iteration. *Journal of machine learning research*, 4(Dec):1107–1149.

Lai, T. L. and Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22.

Lale, S., Azizzadenesheli, K., Anandkumar, A., and Hassibi, B. (2019). Stochastic linear bandits with hidden low rank structure. *arXiv preprint arXiv:1901.09490*.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge university press.

Lazaric, A., Ghavamzadeh, M., and Munos, R. (2010). Finite-sample analysis of lstd. In *ICML-27th International Conference on Machine Learning*, pages 615–622.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, J. M. (2006). *Riemannian manifolds: an introduction to curvature*, volume 176. Springer Science & Business Media.

Levine, N., Zahavy, T., Mankowitz, D. J., Tamar, A., and Mannor, S. (2017). Shallow updates for deep reinforcement learning. *arXiv*.

Levine et al., S. (2016). End-to-end training of deep visuomotor policies. *JMLR*.

Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM.

Li, L., Walsh, T. J., and Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. In *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics (ISAIM-06)*.

Li, Y., Yin, B., and Xi, H. (2011). Finding optimal memoryless policies of pomdps under the expected average reward criterion. *European Journal of Operational Research*, 211(3):556–567.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*.

Lipton, Z. C., Azizzadenesheli, K., Kumar, A., Li, L., Gao, J., and Deng, L. (2016a). Combating reinforcement learning's sisyphean curse with intrinsic fear. *arXiv preprint arXiv:1611.01211*.

Lipton, Z. C., Gao, J., Li, L., Li, X., Ahmed, F., and Deng, L. (2016b). Efficient exploration for dialogue policy learning with bbq networks & replay buffer spiking. *arXiv preprint arXiv:1608.05081*.

Littman, M. L. (1994). Memoryless policies: Theoretical limitations and practical results. In *Proceedings of the Third International Conference on Simulation of Adaptive Behavior : From Animals to Animats 3: From Animals to Animats 3*, SAB94, pages 238–245, Cambridge, MA, USA. MIT Press.

Littman, M. L., Sutton, R. S., and Singh, S. (2001). Predictive representations of state. In *In Advances In Neural Information Processing Systems 14*, pages 1555–1561. MIT Press.

Loch, J. and Singh, S. P. (1998). Using eligibility traces to find the best memoryless policy in partially observable markov decision processes. In *ICML*, pages 323–331.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., and Bowling, M. (2017). Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *arXiv preprint arXiv:1709.06009*.

Madani, O. (1998). On the computability of infinite-horizon partially observable markov decision processes. In *AAAI98 Fall Symposium on Planning with POMDPs, Orlando, FL*.

Madani, O., Hanks, S., and Condon, A. (1999). On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision problems. In *AAAI/IAAI*, pages 541–548.

Maillard, O.-A. and Mannor, S. (2014). Latent bandits. In *Proceedings of the Thirty-First International Conference on Machine Learning (ICML'14)*.

McClelland, J. L., McNaughton, B. L., and O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*.

McCloskey, M. and Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*.

Moldovan, T. M. and Abbeel, P. (2012). Safe exploration in Markov decision processes. In *ICML*.

Murata, M. and Ozawa, S. (2005). A memory-based reinforcement learning model utilizing macro-actions. In *Adaptive and Natural Computing Algorithms*. Springer.

Nadler, B. (2008). Finite sample approximation results for principal component analysis: A matrix perturbation approach. *The Annals of Statistics*, 36(6):2791–2817.

Netrapalli, P., Niranjan, U., Sanghavi, S., Anandkumar, A., and Jain, P. (2014). Non-convex robust pca. In *Advances in Neural Information Processing Systems*, pages 1107–1115.

Ng, A. Y. and Jordan, M. (2000). Pegasus: A policy search method for large mdps and pomdps. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, UAI'00, pages 406–415, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Night, W. (2016). The AI that cut google's energy bill could soon help you. *MIT Tech Review*.

Ortner, P. and Auer, R. (2007). Logarithmic online regret bounds for undiscounted reinforcement learning. *Advances in Neural Information Processing Systems*, 19:49.

Ortner, R. (2013). Adaptive aggregation for reinforcement learning in average reward Markov decision processes. *Annals of Operations Research*, 208(1):321–336.

Ortner, R., Maillard, O.-A., and Ryabko, D. (2014). Selecting near-optimal approximate state representations in reinforcement learning. In Auer, P., Clark, A., Zeugmann, T., and Zilles, S., editors, *Algorithmic Learning Theory*, volume 8776 of *Lecture Notes in Computer Science*, pages 140–154. Springer International Publishing.

Ortner, R. and Ryabko, D. (2012). Online regret bounds for undiscounted continuous reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1763–1771.

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. (2016). Deep exploration via bootstrapped dqn. In *Advances in Neural Information Processing Systems*.

Osband, I., Russo, D., and Van Roy, B. (2013). (more) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*.

Osband, I. and Van Roy, B. (2014a). Model-based reinforcement learning and the eluder dimension. In *Advances in Neural Information Processing Systems*, pages 1466–1474.

Osband, I. and Van Roy, B. (2014b). Near-optimal reinforcement learning in factored mdps. In *Advances in Neural Information Processing Systems*, pages 604–612.

Osband, I., Van Roy, B., and Wen, Z. (2014). Generalization and exploration via randomized value functions. *arXiv*.

Ostrovski, G., Bellemare, M. G., Oord, A. v. d., and Munos, R. (2017). Count-based exploration with neural density models. *arXiv*.

Papadimitriou, C. and Tsitsiklis, J. N. (1987a). The complexity of markov decision processes. *Math. Oper. Res.*, 12(3):441–450.

Papadimitriou, C. H. and Tsitsiklis, J. N. (1987b). The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Peña, V. H., Lai, T. L., and Shao, Q.-M. (2009). *Self-normalized processes: Limit theory and Statistical Applications*. Springer Science & Business Media.

Perkins, T. J. (2002). Reinforcement learning for POMDPs based on action values and stochastic optimization. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI 2002)*, pages 199–204. AAAI Press.

Png, S., Pineau, J., and Chaib-draa, B. (2012). Building adaptive dialogue systems via bayes-adaptive pomdps. *Selected Topics in Signal Processing, IEEE Journal of*, 6(8):917–927.

Porta, J. M., Vlassis, N., Spaan, M. T., and Poupart, P. (2006). Point-based value iteration for continuous pomdps. *Journal of Machine Learning Research*, 7(Nov):2329–2367.

Poupart, P. and Boutilier, C. (2003). Bounded finite state controllers. In Thrun, S., Saul, L. K., and Schölkopf, B., editors, *NIPS*, pages 823–830. MIT Press.

Poupart, P. and Vlassis, N. (2008). Model-based bayesian reinforcement learning in partially observable domains. In *International Symposium on Artificial Intelligence and Mathematics (ISAIM)*.

Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian processes for machine learning*, volume 1. MIT press Cambridge.

Ross, S., Chaib-draa, B., and Pineau, J. (2007). Bayes-adaptive pomdps. In *Advances in neural information processing systems*, pages 1225–1232.

Rubinstein, R. Y. (1969). Some problems in monte carlo optimization. *Ph.D. thesis*.

Rusmevichientong, P. and Tsitsiklis, J. N. (2010). Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411.

Russo, D. and Van Roy, B. (2014a). Learning to optimize via information-directed sampling. *Advances in Neural Information Processing Systems*, pages 1583–1591.

Russo, D. and Van Roy, B. (2014b). Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243.

Schmidhuber, J. (1991). A possibility for implementing curiosity and boredom in model-building neural controllers. In *From animals to animats: SAB90*. Citeseer.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Schweitzer, P. J. and Seidmann, A. (1985). Generalized polynomial approximations in markovian decision processes. *Journal of mathematical analysis and applications*, 110(2):568–582.

Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016). Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv*.

Shani, G., Pineau, J., and Kaplow, R. (2013). A survey of point-based pomdp solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51.

Sharpe, W. F. (1966). Mutual fund performance. *The Journal of Business*.

Shi, G., Shi, X., O'Connell, M., Yu, R., Azizzadenesheli, K., Anandkumar, A., Yue, Y., and Chung, S.-J. (2018). Neural lander: Stable drone landing control using learned dynamics. *arXiv preprint arXiv:1811.08027*.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *Nature*.

Singh, S. P., Jaakkola, T., and Jordan, M. I. (1994). Learning without state-estimation in partially observable markovian decision processes. In *ICML*, pages 284–292. Citeseer.

Sondik, E. J. (1971). *The optimal control of partially observable Markov processes*. PhD thesis, Stanford University.

Song, L., Anandkumar, A., Dai, B., and Xie, B. (2013). Nonparametric estimation of multi-view latent variable models. *arXiv preprint arXiv:1311.3287*.

Spaan, M. T. (2012). Partially observable markov decision processes. In Wiering, M. and van Otterlo, M., editors, *Reinforcement Learning*, volume 12 of *Adaptation, Learning, and Optimization*, pages 387–414. Springer Berlin Heidelberg.

Strens, M. (2000). A bayesian framework for reinforcement learning. In *ICML*.

Sugiyama, M. and Kawanabe, M. (2012). *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT Press.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*.

Sutton, R. S. and Barto, A. G. (1998). *Introduction to reinforcement learning*. MIT Press.

Sutton, R. S., Barto, A. G., Bach, F., et al. (1998). *Reinforcement learning: An introduction*. MIT press.

Tesauro, G. (1995). Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*.

Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE.

Tropp, J. A. (2012). User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434.

Tropp, J. A. (2015). An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230.

Tziortziotis, N., Dimitrakakis, C., and Blekas, K. (2013). Linear bayesian reinforcement learning. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence*.

Valko, M., Munos, R., Kveton, B., and Kocák, T. (2014). Spectral bandits for smooth graph functions. In *International Conference on Machine Learning*, pages 46–54.

Van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double q-learning. In *AAAI*.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.

Vaswani, N. and Narayanamurthy, P. (2017). Finite sample guarantees for pca in non-isotropic and data-dependent noise. In *Communication, Control, and Computing (Allerton), 2017 55th Annual Allerton Conference on*, pages 783–789. IEEE.

Vlassis, N., Littman, M. L., and Barber, D. (2012). On the computational complexity of stochastic controller optimization in pomdps.

Watkins, C. J. and Dayan, P. (1992a). Q-learning. *Machine Learning*.

Watkins, C. J. and Dayan, P. (1992b). Q-learning. *Machine learning*, 8(3-4):279–292.

Williams, J. K. and Singh, S. P. (1998). Experimental results on learning stochastic memoryless policies for partially observable markov decision processes. In Kearns, M. J., Solla, S. A., and Cohn, D. A., editors, *NIPS*, pages 1073–1080. The MIT Press.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.