Robust Structured Prediction for Process Data

by

Xiangli Chen

Thesis submitted in partial fulfillment of the requirements for the degree of the Doctor of Philosophy in Computer Science in the Graduate College of the University of Illinois at Chicago, 2017

Chicago, Illinois

Defense Committee:
Brian D. Ziebart, Chair and Advisor
Piotr J. Gmytrasiewicz
Tanya Y. Berger-Wolf
Byron Boots, Georgia Institute of Technology
Umar Ali Syed, Google
To my parents and sister
ACKNOWLEDGMENTS

It would not have been possible to finish this thesis without the support and encouragement that I received from many people.

First and foremost, I want to express my sincere gratitude to my advisor, Professor Brian D. Ziebart, for his continuous support of my Ph.D study. I appreciate all his contributions of time for questing me, for polishing my knowledge and for improving my ability in thinking, analyzing and solving challenging problems. Thanks to him, my Ph.D work concentrating on artificial intelligence matches my interest. The way how artificial intelligence impacts the world, changes the world and makes the life better makes me full of passion to continue working on it. Besides my advisor, I want to thank the rest of my thesis committee members including Professor Piotr J. Gmytrasiewicz, Professor Tanya Y. Berger-Wolf, Professor Byron Boots and Dr. Umar Ali Syed for providing valuable feedback of my thesis.

I thank my collaborators: Anqi Liu, Dr. Mathew Monfort and Dr. Peter Carr for their discussion, coding and writing in our endeavors together to get our work published. I thank my lab mates: Kaiser Asif, Rizal Fathony, Sima Behpour, Jia Li, Hong Wang, Wei Xing, Chris Schultz, Sanket Gaurav, Andrea Tirinzoni, for maintaining servers, for sharing resources, information and knowledge, for chat, humor and discussions, for drink and food, and for all of the fun we have had during my Ph.D study.
Lastly, I would like to thank my parents who raised me with a love and supported me in all my pursuits. I would like to thank my sister for her love, concern and encouragement. Their support during my Ph.D study is irreplaceable. I would not have made it this far without them.

XC
CONTRIBUTION OF AUTHORS

Chapter 4 presents a published manuscript (Chen and Ziebart, 2015) of a research project, for which I derived and proved most of the theorems, designed and implemented the method, and carried out the experimental work under the guidance of my advisor, Prof. Brian D. Ziebart. I was the primary author. Chapter 5 presents a published manuscript (Chen et al., 2016a) of a research project, for which I completed most of the theoretical and experimental work under the guidance of Prof. Brian D. Ziebart. Dr. Mathew Monfort contributed to processing and running some experimental datasets, and proofreading the manuscript. Anqi Liu contributed to some discussions, drawing some figures and proofreading the manuscript. I was the primary author. Chapter 6 presents a published manuscript (Chen et al., 2016b) of a research project, for which I derived and implemented the method, and carried out the experimental work under the guidance of Prof. Brian D. Ziebart. Dr. Mathew Monfort contributed to some experimental work for comparison and proofreading the manuscript. Dr. Peter Carr contributed to helpful discussions, suggestions, providing the dataset for real experimental validation and proofreading the manuscript. I was the primary author.

Prof. Brian D. Ziebart provided very helpful conjectures, suggestions and feedback throughout the process of getting all of these projects published. I wrote and revised these manuscripts in close collaboration with him.
# TABLE OF CONTENTS

## CHAPTERS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Problem Formulation</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Related Work</td>
<td>6</td>
</tr>
<tr>
<td>1.3 Contributions to Process Prediction</td>
<td>7</td>
</tr>
<tr>
<td>1.4 Thesis Organization and Reader's Guide</td>
<td>10</td>
</tr>
<tr>
<td>2 BACKGROUND</td>
<td>11</td>
</tr>
<tr>
<td>2.1 Information Theory</td>
<td>11</td>
</tr>
<tr>
<td>2.1.1 The Principle of Maximum Entropy</td>
<td>13</td>
</tr>
<tr>
<td>2.1.2 Properties of Entropy and Its Extensions</td>
<td>18</td>
</tr>
<tr>
<td>2.2 Game Theory</td>
<td>23</td>
</tr>
<tr>
<td>2.2.1 The Strategic Form</td>
<td>24</td>
</tr>
<tr>
<td>2.2.2 Solving Games</td>
<td>27</td>
</tr>
<tr>
<td>2.3 Markov Decision Process (MDP)</td>
<td>32</td>
</tr>
<tr>
<td>2.3.1 Infinite-Horizon MDP</td>
<td>35</td>
</tr>
<tr>
<td>2.3.2 Finite-Horizon MDP</td>
<td>41</td>
</tr>
<tr>
<td>2.3.3 Partially Observable MDP (POMDP)</td>
<td>43</td>
</tr>
<tr>
<td>2.4 Reinforcement Learning</td>
<td>46</td>
</tr>
<tr>
<td>2.5 Optimal Control</td>
<td>50</td>
</tr>
<tr>
<td>2.5.1 Linear-Quadratic Regulator (LQR)</td>
<td>51</td>
</tr>
<tr>
<td>2.5.2 Linear-Quadratic-Gaussian System (LQG)</td>
<td>52</td>
</tr>
<tr>
<td>2.6 Covariate Shift</td>
<td>54</td>
</tr>
<tr>
<td>2.6.1 Importance Reweighted Method</td>
<td>56</td>
</tr>
<tr>
<td>3 RELATED WORK</td>
<td>59</td>
</tr>
<tr>
<td>3.1 Direct Estimation</td>
<td>59</td>
</tr>
<tr>
<td>3.1.1 Linear Regression</td>
<td>60</td>
</tr>
<tr>
<td>3.2 Inverse Reinforcement Learning</td>
<td>62</td>
</tr>
<tr>
<td>3.2.1 Maximum Margin Planning</td>
<td>65</td>
</tr>
<tr>
<td>3.2.2 Feature Expectation Matching</td>
<td>67</td>
</tr>
<tr>
<td>3.2.3 Maximum Causal Entropy</td>
<td>68</td>
</tr>
<tr>
<td>4 PREDICTIVE IOC FOR LQG</td>
<td>74</td>
</tr>
<tr>
<td>4.1 Background and Related Work</td>
<td>76</td>
</tr>
<tr>
<td>4.1.1 Linear-Quadratic-Gaussian Control</td>
<td>76</td>
</tr>
<tr>
<td>4.1.2 Inverse Optimal Control</td>
<td>80</td>
</tr>
<tr>
<td>4.1.3 Directed Information Theory</td>
<td>81</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2 Inverse Linear-Quadratic-Gaussian Control</td>
<td>82</td>
</tr>
<tr>
<td>4.2.1 Robust Policy Estimation</td>
<td>82</td>
</tr>
<tr>
<td>4.2.2 A Convex Definition of the LQG Policy Set</td>
<td>84</td>
</tr>
<tr>
<td>4.2.3 Maximum Causal Entropy Estimation</td>
<td>85</td>
</tr>
<tr>
<td>4.2.4 Predictive Inverse LQG Distribution</td>
<td>86</td>
</tr>
<tr>
<td>4.3 Experiments</td>
<td>90</td>
</tr>
<tr>
<td>4.3.1 Synthetic Experiment</td>
<td>91</td>
</tr>
<tr>
<td>4.3.2 Mouse Cursor Experiment</td>
<td>92</td>
</tr>
<tr>
<td>4.4 Summary</td>
<td>97</td>
</tr>
<tr>
<td>5 ROBUST COVARIATE SHIFT REGRESSION</td>
<td>98</td>
</tr>
<tr>
<td>5.1 Related Work</td>
<td>100</td>
</tr>
<tr>
<td>5.1.1 Least Squares Linear Regression</td>
<td>101</td>
</tr>
<tr>
<td>5.1.2 Importance Weighted Linear Regression</td>
<td>101</td>
</tr>
<tr>
<td>5.1.3 Adaptive Importance Weighted Linear Regression</td>
<td>103</td>
</tr>
<tr>
<td>5.1.4 Robust Minimax Learning</td>
<td>103</td>
</tr>
<tr>
<td>5.2 Robust Bias-Aware Regression</td>
<td>104</td>
</tr>
<tr>
<td>5.2.1 Minimax Estimation Formulation</td>
<td>104</td>
</tr>
<tr>
<td>5.2.2 Parameter Estimation</td>
<td>108</td>
</tr>
<tr>
<td>5.2.3 Relation with Other Methods</td>
<td>113</td>
</tr>
<tr>
<td>5.3 Experiments</td>
<td>115</td>
</tr>
<tr>
<td>5.3.1 Datasets</td>
<td>115</td>
</tr>
<tr>
<td>5.3.2 Constructing Dataset with Bias</td>
<td>116</td>
</tr>
<tr>
<td>5.3.3 Density Estimation</td>
<td>118</td>
</tr>
<tr>
<td>5.3.4 Comparison Approaches</td>
<td>119</td>
</tr>
<tr>
<td>5.3.5 Results</td>
<td>121</td>
</tr>
<tr>
<td>5.4 Summary</td>
<td>122</td>
</tr>
<tr>
<td>6 ADVERSARIAL IMITATION LEARNING FRAMEWORK</td>
<td>123</td>
</tr>
<tr>
<td>6.1 Background and Notation</td>
<td>125</td>
</tr>
<tr>
<td>6.2 Problem Definition</td>
<td>126</td>
</tr>
<tr>
<td>6.3 Adversarial Approach</td>
<td>129</td>
</tr>
<tr>
<td>6.3.1 Adversarial Formulation and Properties</td>
<td>129</td>
</tr>
<tr>
<td>6.3.2 Learning and Inference Algorithms</td>
<td>132</td>
</tr>
<tr>
<td>6.4 Experiments</td>
<td>137</td>
</tr>
<tr>
<td>6.4.1 Navigation Across a Grid</td>
<td>137</td>
</tr>
<tr>
<td>6.4.2 Learning Camera Control from Demonstration</td>
<td>141</td>
</tr>
<tr>
<td>6.5 Summary</td>
<td>144</td>
</tr>
<tr>
<td>7 CONCLUSION</td>
<td>146</td>
</tr>
<tr>
<td>7.1 Discussion</td>
<td>146</td>
</tr>
<tr>
<td>7.2 Future Work</td>
<td>148</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>PAGE</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------</td>
</tr>
<tr>
<td>APPENDICES</td>
<td>150</td>
</tr>
<tr>
<td>Appendix A</td>
<td>151</td>
</tr>
<tr>
<td>Appendix B</td>
<td>169</td>
</tr>
<tr>
<td>CITED LITERATURE</td>
<td>174</td>
</tr>
<tr>
<td>VITA</td>
<td>186</td>
</tr>
</tbody>
</table>
LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>116</td>
</tr>
<tr>
<td>II</td>
<td>117</td>
</tr>
<tr>
<td>III</td>
<td>134</td>
</tr>
</tbody>
</table>

I Datasets for empirical evaluation

II Experimental settings

III The payoff matrix for the adversarial IOC prediction game with

\[ \ell(\delta, \delta) = \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t)|\delta, \tau, \hat{\delta}, \hat{\tau}| \right] \text{ and } \psi(\delta) = \omega \cdot \mathbb{E} \left[ \sum_{t=1}^{T} \phi(S_t)|\delta, \tau| \right]. \]
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Bayesian network for linear-quadratic-Gaussian system</td>
</tr>
<tr>
<td>4.2</td>
<td><strong>Left:</strong> Withheld average trajectory log-loss as the observation noise, $\sigma_o$, increases (with fixed state transition dynamic noise $\sigma_d = 0.01$). <strong>Right:</strong> Withheld average trajectory log-loss as the state transition dynamics noise, $\sigma_d$, increases (with fixed observation noise $\sigma_o = 1.0$).</td>
</tr>
<tr>
<td>4.3</td>
<td>Example mouse cursor trajectories terminating at small circle positions exhibiting characteristics of delayed feedback</td>
</tr>
<tr>
<td>4.4</td>
<td>Average trajectory log-loss of: the LQG model with various amounts of delay, $t_0$; the LQR model; Markov models of order 2,3,4</td>
</tr>
<tr>
<td>5.1</td>
<td>Hahn1 taste representing the result of a National Institute of Standards and Technology (NIST) study of the thermal expansion of copper</td>
</tr>
<tr>
<td>5.2</td>
<td>The conditional mean (solid blue line) and 95% confidence interval (CI, dashed dot magenta line) of least squares linear regression (LS), importance weighted least squares (IWLS) and robust bias-aware regression (RBA) via KL-divergence learned via 90 biased source samples (red cross) and evaluated on 118 target datapoints (black point) of the Hahn1 dataset</td>
</tr>
<tr>
<td>5.3</td>
<td>Five plots of the average empirical logloss of seven methods for target datasets with 95% confidence interval with amounts of source data. A bar figure showing empirical log loss on four natural bias datasets</td>
</tr>
<tr>
<td>6.1</td>
<td>Learning to imitate a slower robot capable of walking over barriers</td>
</tr>
<tr>
<td>6.2</td>
<td>Experimental results with 95% confidence interval of various settings of the grid world’s characteristics, including: the degree of stochasticity of the dynamics (top, left); varying amount of cost noise generating the demonstrator’s trajectories (top, center); differences size of the grid world from 5x5 to 15x15 (top, right); different amount of training (test) data (bottom, left); the learner’s dynamics differing from the demonstrator’s (bottom, center); and the introduction of impassible obstacles for the learner (bottom, right)</td>
</tr>
<tr>
<td>6.3</td>
<td>Imitating human camera operator’s pan angle control (the Real trajectory on the left) using a regression approach, maximum margin planning, and our adversarial inverse optimal control method. Average squared loss and absolute loss of the imitator (with 95% mean confidence intervals estimates) are shown in the center and right plots, with maximum margin planning results suppressed due to being significantly worse and off of the presented scale</td>
</tr>
</tbody>
</table>
List of Algorithms

2.1 Policy iteration ................................................................. 40
2.2 Value iteration ................................................................. 41
2.3 Q-learning ................................................................. 50
5.1 Batch gradient descent method for robust bias-aware regression 112
6.1 Double oracle method for adversarial IOC .............................. 135
6.2 Learning algorithm for adversarial IOC .............................. 137
SUMMARY

Processes involve a series of actions performed to achieve a particular result. Developing prediction models for process data is important for many real problems such as human and animal behavior modeling, psychological evaluation, labor hiring cost assessment, stock investment, human robot interaction and so on.

Direct estimation, also known as behavioral cloning, learns a prediction model from training samples using classical supervised learning methods including linear regression, logistic regression, support vector machines, neural networks and so on. These methods, which are optimized by penalizing deviation of each single next variable from training samples are not adaptive to general process prediction tasks, especially decision control settings. Inverse reinforcement learning (IRL), also known as inverse optimal control (IOC), is a structured prediction approach that jointly estimates the response of the process to all possible scenarios. IRL attempts to recover a reward function that rationalizes the process strategy under the assumption that training samples are generated by performing a (near) optimal strategy with respect to an unknown reward function which describes the purpose of the process. Unfortunately, this assumption often doesn’t hold in practice. Early work of IRL based on this strong assumption of optionally leads to prediction methods with unstable performance. Recent work of IRL based on the principle of maximum causal entropy derives the best probability distribution estimate of process behavior under worst case assumptions to provide robust prediction performance in real applications. However, this approach assumes the process environment is fully observable.
Furthermore, it assumes the process settings of learning and prediction share the same properties. In addition, the evaluation measure of the maximum causal entropy approach is restricted to evaluating using the logarithmic loss.

Our contribution presented in this thesis first tractably extends the principle of maximum causal entropy to certain partially observable environments. More specifically, we develop IRL methods for the linear-quadratic-Gaussian system, a well known optimal control problem with partial observability. Furthermore, we investigate process prediction problems under non-stationary settings. One form of this problem is known as covariate shift, where the input distributions for training and testing are different while the mappings from input to output are the same. We propose a robust approach to deal with covariate shift for linear regression as a significant first step to deal with covariate shift for general process prediction tasks. Finally, we introduce a general framework for imitation learning, an important process prediction task where a learner attempts to imitate a demonstrator’s behavior from observed demonstration. Our framework enables learning for general evaluation measures and different capabilities between the learner and demonstrator. We demonstrate the effectiveness and show the benefits of our approaches on both synthetic and real datasets.
CHAPTER 1

INTRODUCTION

Process

“A series of actions or events performed to make something or achieve a particular result, or a series of changes that happen naturally.”

—–Cambridge Dictionary

Developing prediction models for process data is important for many real problems. Human and animal behavior modeling has been widely investigated as a process prediction problem. Examples include modeling bee foraging (Montague et al., 1995) in uncertain environments, songbird vocalization (Doya and Sejnowski, 1995) and context-dependent route preference of taxi drivers (Ziebart et al., 2008b). One of the main goals of psychology is to predict the behavior and mental processes of others (Sutton and Barto, 1998). Process prediction in economics has been well studied with numerous applications, such as assessing the labor hiring cost by examining a firm’s hiring strategy over time (Rust, 1992). Economic forecasting, the process of making predictions about the economy, remains an important decision-making tool for business and government as they formulate financial policies and strategies. Process prediction is also of key importance in artificial intelligence. For example, human robot interaction requires a robot to be able to infer a user’s intention and makes optimal actions over the interaction process. One human robot interaction application (Balakrishnan, 2004) concerns whether robots can
efficiently and safely navigate around people, as well as user interfaces that autonomously improve a user’s task efficiency. Each requires highly accurate process prediction. Recent progress of process prediction work has led to some notable achievements such as helicopter aerobatic maneuvers (Abbeel et al., 2007) and autonomous vehicle navigation (Silver et al., 2010).

Learning a good prediction model from process data is a challenging machine learning and artificial intelligence task. The environment usually is complex, and the collected process data is often noisy. It may only contain limited information of the environment, meaning that the environment is partially observable. The process setting to which a prediction model is deployed may be different from the one under which the data is collected, i.e., a non-stationary setting. One form of this problem is known as covariate shift where the mapping from inputs to output is shared during learning and prediction, but the distribution of input can vary. Learning from demonstration, also known as imitation learning, is an important process prediction task where a learner attempts to mimic a demonstrator’s behavior from demonstrated samples. In practice, the embodiment of a learner can be significantly different from the demonstrator. For example, a robot’s joint and torque limits differ from those of a human demonstrator. We refer to the resulting problem as imitation learning with embodiment transfer. In addition, due to the variety of imitation learning tasks, prediction models enabling various evaluation measures are required.

Our contributions presented in this thesis focus on developing robust prediction models that allow partially observable continuous environments (Chen and Ziebart, 2015),
deal with covariate shift (Chen et al., 2016a), and enable various imitation learning evaluation measures and embodiment transfer (Chen et al., 2016b). Robustness is provided by deriving the best probability distribution estimation under the worst case subject to matching known properties of the real probability distribution, a minimax estimation approach (Topsøe, 1979; Grünwald and Dawid, 2004). We demonstrate the effectiveness and benefits of our robust approaches on both synthetic and real datasets.

1.1 Problem Formulation

In artificial intelligence (Russell, 2009), a process prediction problem often arises within an interaction process between an agent and its environment. An agent is an entity that perceives its environment and acts upon that environment. The choice of an action depends on the information it senses from its environment, and the environment usually is updated after the action is executed. The environment may be fully observable so that the agent can sense all information that is relevant to its choice of action. Otherwise, it is partially observable. A state contains all of the information that are necessary to choose an action. An environment is deterministic if the next state of that environment is completely determined by the current state and the action being executed. Otherwise, it is stochastic. Moreover, a continuous or discrete context of a predicting process could apply to the states of the environment, the actions of the agent, or both.

Throughout this thesis, we assume that an agent producing a process starts with an initial state $s_1$, executes its first action $a_1$, and then perceives a new state $s_2$ followed by a new action $a_2$ and so on that updating states alternates with executing actions. In other words,
a process generates the following sequence: \( s_1, a_1, \ldots, a_{T-1}, s_T \) if it ends at time \( T \). Roughly speaking, a **process** is a series of actions performed to achieve a particular result. A strategy of choosing actions is known as a **policy**. The **state transition dynamics** of a process indicates how states are updated. A policy and the state transition dynamics specify how a process is generated. Most of our work presented in this thesis concerns estimating a policy. We give a formal definition of **process prediction** concerning estimating a policy from given training samples in definition 1.1.

**Definition 1.1.** A **process prediction** problem is an estimation task that given a list of training samples \( \{s_1, a_1, \ldots, s_{T_n-1}, a_{T_n-1}, s_{T_n}\}_{n=1}^N \) generated from performing one or a few unknown policies on a process under a setting with a state transition dynamics \( \tau_t(s_t|s_{1:t-1}, a_{1:t-1}) \) and constraint \( s_t \in S_t, a_t \in A_t \), we want to estimate a policy \( \hat{\pi}_t(a_t|s_{1:t}, a_{1:t-1}) \) for a process under a setting with a state transition dynamics \( \hat{\tau}_t(s_t|s_{1:t-1}, a_{1:t-1}) \) and constraint \( s_t \in \hat{S}_t, a_t \in \hat{A}_t \) with respect to a performance evaluation method.

Note that by definition, a state is sufficient to choose an action. But, in practice, information collected for updating states may be noisy. So definition 1.1 treats the policy in a more general form so that choosing an action depends on all previous states and actions.

As for partially observable environments, states \( s_t \) are assumed unknown and observations \( o_t \) are given. Besides the state transition dynamics, an **observation dynamics** \( p_t(o_t|s_{1:t}, a_{1:t}) \) specifies how observations are distributed conditioned on previous states and actions. A policy under a partially observable environment is the strategy of choosing an action depending on previous observations and actions \( \pi_t(a_t|o_{1:t}, a_{1:t-1}) \).
Moreover, it is common that training samples only contain states. One situation that knowing states is sufficient for a process prediction problem is when the dynamics of that process is deterministic. In this case, if states are numerical values, actions are simply the deviations of the states.

Due to the variety of the purpose of process prediction tasks, there are various ways to evaluate an estimated policy so as to compare the performance of various process prediction methods. We may want a robot to grab a cup, then the evaluation method is simply to check if an estimated policy deployed to that robot can successfully grab a cup. A more sophisticated way assumes that a process is performed so as to maximize expected long term reward. The reward value is specified by a reward function. A prediction model is then developed to recover a reward function assuming the given training samples are generated by a policy which maximizes the expected long term reward with respect to that unknown reward function. However, since the real reward function is unknown, evaluating an estimated policy by measuring the value of the expected long term reward is not possible. A standard machine learning evaluation approach is to define a loss function to evaluate an estimated policy on a list of test samples. Expected log-loss measures the distance of two distributions motivated by information theory on encoding and sending messages (Topsøe, 1979; Grünwald and Dawid, 2004) (Section 2.1.1). Empirical log-loss is often used to measure how likely a list of test samples are generated by a policy. A 0 – 1 loss might be used if we are concerned with the loss of classifying enumerable values such as modeling daily activities of human beings. As for learning camera control from demonstrated trajectories of cameras’ pan angles, a square loss
or absolute loss of a camera’s pan angle might be used if we want to learn a robotic camera to mimic these trajectories.

1.2 Related Work

A simple and straightforward way to formulate a process prediction task is to treat it as a classical supervised learning problem known as direct estimation or behavior cloning. But, direct estimation (Pomerleau, 1989) such as logistic regression for discrete environment or linear regression for continuous environment, doesn’t consider process prediction in its entirety. In addition, direct estimation using classical supervised learning methods that penalize the deviation from the training samples is not adaptive to many process prediction problems especially the ones under decision control settings. For example, if the state transition dynamics change between training time and testing time, direct estimation will result in poor performance under various evaluation methods.

A more natural and succinct way for developing process prediction models is to consider a process as a series of actions performed to achieve a particular result such as maximizing expected long term reward or minimizing expected long term cost. Inverse reinforcement learning (IRL) (Kalman, 1964; Boyd et al., 1994a; Russell, 1998; Ng et al., 2000), also known as inverse optimal control (IOC), attempts to recover a reward function to rationalize the action taking strategy generalized in given training samples assuming these samples are generated by an optimal policy with respect to an unknown reward function which describes the purpose of a process. However, this optimality assumption often is not true for many real applications. Early work on IRL (Ng et al., 2000; Abbeel and Ng, 2004; Abbeel et al., 2007; Ratliff et al., 2006)
based on this assumption results in unstable prediction performance under various evaluation methods. Recent work on IRL based on the principle of maximum causal entropy (Ziebart et al., 2008a; Ziebart et al., 2010; Ziebart et al., 2012; Ziebart et al., 2013) which derives the best probability distribution estimation of policy under the worst case has demonstrated robust performance in real applications evaluated on log-loss and becomes one of the most preferred IRL methods for process prediction tasks.

1.3 Contributions to Process Prediction

In practice, the process environment is complex and noisy. Thus process prediction concerning estimating a policy intrinsically is challenging. Predictive inverse optimal control (IOC) is a powerful approach for estimating control policy from observed control demonstration, and its usefulness has been established in a number of large-scale sequential decision settings characterized by complete state observability (Ramachandran and Amir, 2007; Ziebart et al., 2008a; Babes et al., 2011; Neu and Szepesvári, 2012). However, many real decisions are made in situations where the state is not fully known to the agent making decisions. Though extensions of predictive inverse optimal control to partially observable Markov decision processes (POMDP) have been developed (Choi and Kim, 2011), their applicability has been limited by the complexities of inference in those representations. Our work (Chen and Ziebart, 2015) extends predictive inverse optimal control (IOC) to the linear-quadratic-Gaussian control setting (LQG), a well known optimal control problem with partially observable continuous setting. Our prediction model for the LQG setting is a minimax estimation approach subject to feature expectation constraint where the control policies are derived from minimax expected
causal log loss which can be used for measuring the control polices estimation performance over control process setting. We also establish close connections between optimal control laws for this setting and the probabilistic predictions under our approach. We demonstrate the effectiveness and benefit in estimating control policies that are influenced by partial observability on both synthetic and real datasets.

Most of process prediction work assumes that the training samples are representative for learning a prediction model. But, this is often not the case in many real applications. The process setting to which a prediction model is deployed may be different from the one under which the training samples are collected, a non-stationary process setting. For example, driving samples collected from suburbs may not be suitable to be directly used to learn a prediction model for cities. Driving habits in cities may be much more aggressive. One form of this problem, where the input distribution of samples for training is different from the ones for prediction, is known as covariate shift. However, existing methods for dealing with covariate shift attempt to re-weight the training data samples to better represent the prediction domain (Shimodaira, 2000), but this introduces strong inductive biases that are highly extrapolative (Cortes and Mohri, 2014) and can often err greatly in practice. Our robust prediction approach (Chen et al., 2016a) built on minimax relative log-loss (Cover and Thomas, 2006) using a baseline distribution embraces the uncertainty resulting from sample selection bias by producing regression models that are explicitly robust to it. The robust covariate shift regression model is a direct estimation method which may not adapt to general process prediction problems especially the ones under decision control settings. However, we consider it to be the significant first step
to deal with covariate shift under process settings. More sophisticated robust approaches for
general process prediction tasks where covariate shift exists may be derived motivated by robust
covariate shift regression. We demonstrate the benefit of our robust approach on a number of
regression tasks with both synthetically created and naturally existing covariate shift datasets.

Imitation learning is an important process prediction problem where a learner attempts to
imitate a demonstrator’s behavior from observed demonstration. Inverse optimal control (IOC)
is a powerful model for imitation learning, and it has been successfully applied to some imitation
learning tasks (Abbeel et al., 2007; Chung and Huang, 2010). However, earlier work on building
IOC methods for imitation learning either results in unstable prediction performance (Abbeel
and Ng, 2004; Ratliff et al., 2006) or limits to specific loss functions (Ziebart et al., 2010) that
makes them incapable of being effective in a broader range of learning tasks. In addition, most
of the work assumes the learner and the demonstrator share the same capabilities, which often
violates the settings of many learning tasks. For example, the joint and torque limits of a
robotics learner are different from a human demonstrator’s joint limits (Alissandrakis et al.,
2002; Nehaniv and Dautenhahn, 2002). We develop a general framework (Chen et al., 2016b)
for inverse optimal control that enables learning for more general imitative evaluation measures
and differences between the capabilities of the demonstrator and those of the learner when
they have different embodiments. Our formulation based on a minimax estimation approach
takes the form of a two-person zero-sum game (Von Neumann and Morgenstern, 2007; Straffin,
1993) between a learner attempting to minimize an imitative loss measure, and an adversary
attempting to maximize the loss by approximating the demonstrated examples in limited ways.
We establish the consistency and generalization guarantees of this approach and illustrate its benefit on real and synthetic imitation learning tasks.

1.4 Thesis Organization and Reader’s Guide

This thesis is organized into seven chapters: Introduction, Background, Related Work, Predictive IOC for LQG, Robust Covariate Shift Regression, Adversarial Imitation Learning Framework, and Conclusion. Chapter 2 gives brief introduction of some background knowledge that serves as a foundation to our work, including information theory, game theory, Markov decision process, reinforcement learning, optimal control and covariate shift. Chapter 3 reviews related work on developing process prediction methods including direct estimation and inverse reinforcement learning. Chapter 4, Chapter 5 and Chapter 6 present our work on developing robust prediction models for process prediction problems that allow partially observable continuous environments, deal with covariate shift, and enable general imitation learning losses and embodiment transfer respectively. Finally, we give the conclusion and discuss the future work of the thesis in Chapter 7.
CHAPTER 2

BACKGROUND

This chapter introduces the background knowledge related to our work on developing robust process prediction methods. The first section, 2.1, introduces information theory to provide the motivation and formulation of robust estimation via the intuition of maximizing uncertainty or minimizing the worst case log-loss. Game theory, discussed in section 2.2, provides a theoretical foundation for our work on formulating a probability estimation problem as a two-person zero-sum game. Section 2.3 introduces the Markov Decision Process (MDP) model where both actions and states are assumed to have a first-order Markov property, and also describes dynamic programming methods for MDPs. The Partially Observable MDP (POMDP) for partially observable environments is also discussed. Reinforcement learning, introduced in section 2.4, motivates the formulation of inverse reinforcement learning. Section 2.5 gives brief introduction of two important optimal control models that are the linear quadratic regulator (LQR) and the linear quadratic Gaussian system (LQG). Finally, we introduce covariate shift in section 2.6, a non-stationary process setting and also the importance reweighting method, for dealing with covariate shift.

2.1 Information Theory

Information theory was founded by Shannon’s work (Shannon, 1948) where he showed that under reasonable assumptions, there is a quantity, known as entropy, measuring the amount
of information or uncertainty of a random process. E.T. Jaynes (Jaynes, 1957) advocated for the use of maximum entropy for probability distribution estimation with the thinking that maximizing entropy is maximizing uncertainty leading to a motivation of robust estimation. Graham Wallis (Jaynes and Bretthorst, 2003) showed another motivation of applying maximum entropy to Jaynes in 1962 by constructing a random probability assignment experiment, known as the Wallis derivation where the most likely probability distribution of the random experiment converges to the one maximizing the entropy. In information theory (Shannon, 1948; Cover and Thomas, 2006), expected log-loss of two probability distributions measures the amount of the expected number of bits per symbol required to describe a random process encoding by another one. We can interpret it by saying that expected log-loss measures the distance between two probability distributions. It gives us an intuition that expected log-loss between an estimated probability distribution and the real probability distribution measures the goodness of the estimated one. The minimax approach (Topsøe, 1979; Grünwald and Dawid, 2004) formulates the probability estimation problem as selecting the best estimation under the worse case with respect to the expected log loss leading to the formulation of maximum entropy. Maximum uncertainty, the random assignment derivation and the minimax approach establish the robustness of the probability distribution estimation via maximizing entropy, known as the principle of maximum entropy. The principle of maximum entropy and its extensions for probability distribution estimation have been successfully applied to many problems including statistical mechanics, natural language processing, economics, and ecology (Dudik and Schapire, ...
2006) with impressive performance. The detail of the principle of maximum entropy is presented in section 2.1.1. Section 2.1.2 discusses the important properties of entropy and its extensions.

### 2.1.1 The Principle of Maximum Entropy

Suppose there is a set of possible events with a probability distribution \((p_1, p_2, ..., p_n)\). Assume there exists a numerical measure denoted by \(H(p_1, p_2, ..., p_n)\) mapping from the amount of uncertainty represented by the probability distribution \((p_1, p_2, ..., p_n)\) to real number. Shannon claimed that it was reasonable of \(H(p_1, p_2, ..., p_n)\) to satisfy the following conditions:

1. \(H\) is a continuous function of \(p_i\). A small change in probability distribution should result in a small change in the amount of uncertainty.
2. \(H\left(\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n}\right)\) is a monotonic increasing function of \(n\) that more possibilities has more uncertainty.
3. \(H\) should be consistent that it should give same value for every possible ways of working out it. If an random event be broken down into two successive random events, the value of \(H\) of that random event should be the weighted sum of the values of \(H\) of the two successive ones.

**Theorem 2.1.** The only \(H\) satisfying the three conditions above is of the form (Shannon, 1948):

\[
H = -K \sum_{i=1}^{n} p_i \log p_i.
\]

*The constant \(K\) merely amounts to the choice of a unit of measure.*
From this measuring uncertainty viewpoint, maximizing the entropy $H$ (Jaynes, 1957) by matching known constraints prevents us as much as possible from imposing any personal preference or unsupported evidence in probability distribution estimation.

The Wallis derivation (Jaynes and Bretthorst, 2003) assumes the probability distribution estimation for a random process is to assign probability quantities to possible outcomes. Suppose there are $m$ possible outcomes and some integer $n \gg m$ that the probability is divided into $n$ little quanta with the magnitude $\delta = 1/n$. We randomly assign these quanta to the $m$ possible outcomes. After assigning the $n$ quanta, the distribution of the probability quantity of the $m$ outcomes is

$$p_i = n_i \delta = \frac{n_i}{n}, i = 1, 2, ..., m.$$  

The probability of an assignment is

$$m^{-n}.$$  

The number of assignments matching the same distribution of the probability quantity is

$$C_n^{n_1} C_{n-n_1}^{n_2} \cdots C_{n_m}^{n_m} = \frac{n!}{n_1! \cdots n_m!}.$$  

So the probability of a probability quantity distribution of an assignment experiment is

$$m^{-n} C_n^{n_1} C_{n-n_1}^{n_2} \cdots C_{n_m}^{n_m} = m^{-n} \frac{n!}{n_1! \cdots n_m!}.$$
The most likely probability quantity assignment distribution is

$$\arg\max_{\left\{ \frac{n}{m} \right\}} m^{-n} \frac{n!}{n_1!...n_m!} = \arg\max_{\left\{ \frac{n}{m} \right\}} \frac{n!}{n_1!...n_m!}.$$

Given $n$, this assignment experiment assumes the magnitude of probability quanta is $1/n$, but probability mapping from a sample space to real number is a continuous quantity. We investigate the most likely probability quantity assignment distribution as $n$ goes to infinite.

Let

$$W = \frac{n!}{n_1!...n_m!}.$$

Since $n$ is fixed,

$$\arg\max_{\left\{ \frac{n}{m} \right\}} \frac{1}{n} \log W = \arg\max_{\left\{ \frac{n}{m} \right\}} \log W.$$

Using Stirling’s approximation

$$\log(n!) = n \log(n) - n + \sqrt{2\pi n} + \frac{1}{12n} + O\left(\frac{1}{n^2}\right).$$

So we have

$$\lim_{n \to \infty} \left(\frac{1}{n} \log W\right) = -\sum_{i=1}^{n} p_i \log p_i = H(p_1, p_2, ..., p_m).$$

So

$$\arg\max_{\left\{ p_i \right\}} \lim_{n \to \infty} \left(\frac{1}{n} \log W\right) = \arg\max_{\left\{ p_i \right\}} H(p_1, p_2, ..., p_m).$$
The most likely probability quantity assignment distribution is the one that maximizes $H$. Furthermore, we can slightly modify the assignment procedure to match any required probability distribution constraint. After assigning all quanta, we check if the probability quantity assignment distribution satisfies the constraint. If not, we reject it and try again until the constraint is satisfied. Let $N$ to be the number of all assignments the probability quantity assignment distribution of which satisfies the constraint. The probability of a satisfiable probability quantity assignment distribution is

$$N^{-1} \frac{n!}{n_1!...n_m!},$$

Let $\mathcal{C}$ to be the set of satisfiable probability quantity assignment distributions. The most likely one within $\mathcal{C}$ is

$$\arg \max_{\{\frac{n}{m}\} \in \mathcal{C}} N^{-1} \frac{n!}{n_1!...n_m!} = \arg \max_{\{\frac{n}{m}\} \in \mathcal{C}} \frac{n!}{n_1!...n_m!}.$$ 

As $n$ goes to infinite

$$\arg \max_{\{\frac{n}{m}\} \in \mathcal{C}} \lim_{n \to \infty} \left(\frac{1}{n} \log \frac{n!}{n_1!...n_m!}\right) = \arg \max_{\{\frac{n}{m}\} \in \mathcal{C}} H(p_1, p_2, ..., p_m),$$

the most likely probability quantity assignment distribution which satisfies the required constraint is the one that maximizes the entropy $H$ under the constraint.

In information communication theory (Shannon, 1948; Cover and Thomas, 2006), the length of the optimal prefix-free encoding for a symbol denoted by $x$ with the sending probability distribution $p(x)$ is close to $-\log p(x)$, known as log loss. Entropy $H = \mathbb{E}_p[-\log p(x)]$ provides
a lower bound of the amount of the expected number of bits per symbol needed to describe the random symbol sending process:

\[ \mathbb{E}_p[-\log p(x)] \leq \mathbb{E}_p[-\log q(x)] \]

where \( q(x) \) is any valid probability distribution. \( \mathbb{E}_p[-\log q(x)] \), known as the expected log-loss, gives an intuition about measuring the distance of two distributions \( p \) and \( q \). Assuming \( p \) is the real distribution, the min max approach [Topsøe, 1979; Grünwald and Dawid, 2004] attempts to find an estimator \( q^* \) that minimizes the worst case expected log loss:

\[ q^* = \arg \min_{q \in \mathcal{A}} \max_{p \in \mathcal{P}} \mathbb{E}_p[-\log q(X)] \]

where \( \mathcal{A} \) is the set of valid real distributions, and \( \mathcal{P} \) is the set of valid estimators. The robustness of the minimax approach is built by the following inequality:

\[ \mathbb{E}_p[-\log q^*(X)] \leq \min_{q \in \mathcal{A}} \max_{p \in \mathcal{P}} \mathbb{E}_p[-\log q(X)]. \]

The minimax expected log loss provides the least upper bound for the estimator \( q^*(x) \). Although the real distribution \( p \) is unknown, choosing \( q^* \) to be the estimator of \( p \) guarantees that the
expected log loss will not exceed the minimax one. The connection to maximum entropy is built by strong duality [Boyd and Vandenberghe, 2004] in the case that $\Gamma$ is closed and convex:

$$\max_{p \in \Gamma} H(p) = \max_{p \in \Gamma} \min_{q \in A} E_p[- \log q(X)] = \min_{q \in A} \max_{p \in \Gamma} E_p[- \log q(X)].$$

Let

$$p^* = \arg \max_{p \in \Gamma} H(p),$$

strong duality has the following property:

$$p^* = q^*.$$

So the estimator maximizing the entropy is the one which minimizes the worst case expected log-loss. Maximum entropy and its extensions [Wainwright and Jordan, 2008] has been applied to derive many well known exponential family distributions (e.g., Gaussian, exponential, Laplacian, logistic regression and conditional random field) by incorporating various known constraints.

2.1.2 Properties of Entropy and Its Extensions

This section provides a brief overview of the definition and properties of entropy and its extensions based on Cover & Thomas’s textbook [Cover and Thomas, 2006] which the reader can refer to for more detail. We assume a prefix-free and binary encoding method when we introduce information theory material throughout the thesis.
Definition 2.1. The **Entropy** of a discrete random variable $X$ with distribution $p(x)$ is

$$H(X) = - \sum_{x \in X} p(x) \log p(x) = \mathbb{E}_p[- \log p(X)].$$

Conventionally, $0 \log 0 = 0$, since $\lim_{x \to 0} x \log x = 0$. (let $t = \frac{1}{x}$ and consider $t \to \infty$)

Entropy provides a lower bound of the expected number of bits per symbol or expected code-word length required to encode a random variable.

**Theorem 2.2.** Let $L^*$ be the minimum expected number of bits per symbol or code-word length required to encode a random variable $X$. Then

$$H(X) \leq L^* \leq H(X) + 1.$$ 

Entropy provides a tight bound of the expected code-word length required to encode a random variable. As discussed in section 2.1.1, entropy also can be interpreted as a quantity measuring the amount of uncertainty or information of a random variable. Entropy is a concave function of a probability distribution, and its maximum is obtained when the distribution is evenly spread.

**Lemma 1.**

$$H(X) \geq 0$$
Definition 2.2. The **Joint Entropy** of two random variable \(X\) and \(Y\) is

\[
H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(x, y) = \mathbb{E}_p[\log p(X, Y)].
\]

Definition 2.3. The **Conditional Entropy** of a random variable \(Y\) given \(X\) is

\[
H(Y | X) = \sum_{x \in X} p(x) H(Y | X = x).
\]

Note.

\[
H(Y | X = x) = - \sum_{y \in Y} p(y | x) \log p(y | x)
\]

\[
H(Y | X) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(y | x) = \mathbb{E}_p \log p(Y | X)
\]

The conditional entropy measures the amount of the expected number of bits per symbol needed to describe a random variable \(Y\) given the value of another random variable \(X\).

Theorem 2.3.

\[
H(X, Y) = H(X) + H(Y | X)
\]

Corollary 2.1.

\[
H(X, Y | Z) = H(X | Z) + H(Y | X, Z)
\]

Theorem 2.4. **Chain rule for Entropy** is

\[
H(X_1, X_2, \cdots, X_n) = \sum_{i=1}^{n} H(X_i | X_1, X_2, \cdots, X_{i-1}).
\]
Definition 2.4. Cross Entropy between two distribution $p$ and $q$ of a random variable $X$ is
\[
H(p, q) = -\sum_{x \in X} p(x) \log q(x).
\]

It measures the amount of the expected number of bits per symbol required to describe a random variable $X$ with a distribution $p$ encoding by another distribution $q$. As discussed in section 2.1.1, cross entropy, also known as expected log loss, measures the distance of the two distributions $p$ and $q$. The minimum of cross entropy is obtained when $q$ equals to $p$. It is often used to evaluate the goodness of a probability estimation method where $q$ is the estimator of the real distribution $p$.

Definition 2.5. The Relative Entropy, also known as Kullback-Leibler Divergence, between two distributions $p$ and $q$ of a random variable $X$ is
\[
D(p||q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}.
\]

Note.
\[
D(p||q) = H(p, q) - H(p)
\]

Theorem 2.5.
\[
D(p||q) \geq 0
\]

with equality iff $p(x) = q(x)$ for all $x$. 
The relative entropy measures the additional amount of the expected number of bits per symbol required to describe a random variable $X$ with a distribution $p$ encoding by another distribution $q$. It can evaluate the encoding inefficiency of assuming that the distribution of a random variable is $q$ while the true distribution is $p$.

**Definition 2.6. Differential Entropy** $h(X)$ of a continuous random variable $X$ given a density function $f(x)$ is

$$h(X) = -\int_S f(x) \log f(x) dx$$

where $S$ is the support set of $X$.

**Definition 2.7. Conditional Differential Entropy** of a continuous random variable $Y$ given another continuous one $X$ is

$$h(X|Y) = -\int f(x,y) \log f(x|y) dx dy.$$  

**Note.** In general $f(x|y) = f(x,y)/f(y)$,

$$h(X|Y) = h(X,Y) - h(Y).$$

**Theorem 2.6. Chain rule for Differential Entropy is**

$$h(X_1, X_2, \cdots, X_n) = \sum_{i=1}^n h(X_i|X_1, X_2, \cdots, X_{i-1}).$$
Definition 2.8. The Relative Entropy or Kullback-Leibler Divergence, between two densities \( f(x) \) and \( g(x) \) is

\[
D(f||g) = \int f(x) \log \frac{f(x)}{g(x)}.
\]

Theorem 2.7.

\[
D(p||q) \geq 0
\]

with equality iff \( f(x) = g(x) \) almost everywhere.

2.2 Game Theory

Games are characterized as interaction processes among a number of players or decision makers aiming to maximize individual benefit or minimize individual loss by taking actions under uncertain conditions (Ferguson, 2014). Game theory has very broad applications in many areas ranging from entertainment, economic applications, politics, to biology. Game theory is a large subject. In this section, we restrict our attention to the topic relating to our imitation learning work that is two-person zero sum games. In such a game, one player wins what the other player loses. The theory of two-person zero-sum games are founded by Von Neumann’s work in the late 1920s [Neumann, 1928, Von Neumann and Morgenstern, 2007]. The material about two-person zero-sum games presented in this section is a summary of Thomas S.Ferguson’s textbook on this topic (Ferguson, 2014) which readers can refer to for more details.
2.2.1 The Strategic Form

Strategic form is one of the main mathematical models of games. In strategic form, players choose strategies by considering other players’ possible strategies. Strategies are chosen and revealed simultaneously, and the game ends with players receiving some payoff. There is an assumption that payoffs can be replaced with numerical values. The mathematical and philosophical justification behind this assumption is discussed in utility theory (Neumann et al., 1944). We say that a game in strategic form is zero-sum if the sum of payoffs players receive is zero no matter what strategies they choose. A two person zero-sum game has only two players denoted by player I and II where the player I wins what the player II loses.

Definition 2.9. The strategic form of a two-person zero-sum game can be described as a triplet $(X,Y,A)$, where

- $X$: a nonempty set of strategies of Player I,
- $Y$: a nonempty set of strategies of Player II,
- $A: X \times Y \to \mathbb{R}$ is a function mapping pairs of strategies to real values.

Elements of $X$ or $Y$ are known as pure strategies. A combination of pure strategies in various proportions at random is called a mixed strategy. A strategy is called an equalizing strategy if it produces the same average winnings no matter what its opponent does. If both $X$ and $Y$ are finite sets, $(X,Y,A)$ is known as a finite game.

Theorem 2.8. (The Mini-max Theorem) Given a finite two-person zero-sum game,

- there is a numerical value $V$,
• no matter what II does, player I always has a mixed strategy such that I’s average gain is at least $V$, and

• no matter what I does, player II always has a mixed strategy such that II’s average loss is at most $V$.

$V$ is known as the value of the game.

If $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, we define the game matrix or payoff matrix as

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$$

where $a_{ij} = A(x_i, y_j)$ is the value player I wins or player II loses by choosing the pure strategies $x_i$ and $y_j$ respectively. We can denote a mixed strategy by its probability distribution since the latter can uniquely represent the former. If $p = (p_1, p_2, \ldots, p_m)^T$ is the mixed strategy of player I, and $q = (q_1, q_2, \ldots, q_n)^T$ is the mixed strategy of player II, the average payoff to I is $p^T A p = \sum_{i=1}^m \sum_{j=1}^n p_i a_{ij} q_j$. For a matrix game with $m \times n$ matrix $A$, and $V$ is the value of the game, a strategy $p = (p_1, \ldots, p_m)^T$ is optimal for player I if and only if

$$\sum_{i=1}^m p_i a_{ij} \geq V \ \forall j \in \{1, \ldots, n\}.$$
For player II, a strategy $q = (q_1, \cdots, q_n)^T$ is optimal if and only if

$$\sum_{j=1}^{n} a_{ij}q_j \leq V \quad \forall i \in \{1, \cdots, m\}.$$ 

The average payoff $\sum_i \sum_j p_i a_{ij} q_j$ is exactly $V$ when both strategies are optimal. Here is the short proof:

$$V = \sum_{j=1}^{n} Vq_j \leq \sum_{j=1}^{n} \left( \sum_{i=1}^{m} p_i a_{ij} \right) q_j = \sum_{i=1}^{m} \sum_{j=1}^{n} p_i a_{ij} q_j = \sum_{i=1}^{m} p_i \left( \sum_{j=1}^{n} a_{ij} q_j \right) \leq \sum_{i=1}^{m} p_i V = V.$$ 

It is easy to conclude that the value of game $V$ is unique.

**Theorem 2.9. (The Equilibrium Theorem)** Suppose a game has a $m \times n$ matrix $A$ and a game value $V$, and player I and player II have the optimal strategies $p = (p_1, p_2, \cdots, p_m)^T$ and $q = (q_1, q_2, \cdots, q_n)^T$ respectively. Then

$$\sum_{j=1}^{n} a_{ij}q_j = V \quad \forall i \in \{1, \cdots, m\} \text{ where } p_i > 0$$

and

$$\sum_{i=1}^{m} p_i a_{ij} = V \quad \forall j \in \{1, \cdots, n\} \text{ where } q_j > 0.$$ 

We can interpret the Equilibrium Theorem by saying that if there exists an optimal strategy for player I that gives positive probability to the $i$th pure strategy, every optimal strategy of player II will give the value of game to player I if he uses the $i$th pure strategy. The similar argument holds for player II.
2.2.2 Solving Games

Let \( X^* \) and \( Y^* \) denote the sets of mixed strategies of player I and II of a finite two-person zero-sum game, \((X, Y, A)\) with a \( m \times n \) matrix \( A \) respectively that

\[
X^* = \{ p = (p_1, \cdots, p_m)^T : p_i \geq 0, \forall i \in \{1, \cdots, m\}, \text{ and } \sum_{i=1}^{m} p_i = 1 \},
\]

\[
Y^* = \{ q = (q_1, \cdots, q_n)^T : q_j \geq 0, \forall j \in \{1, \cdots, n\}, \text{ and } \sum_{j=1}^{n} q_j = 1 \}.
\]

For any \( p \in X^* \) and \( q \in Y^* \), the average payoff to player I is

\[
\sum_{i=1}^{m} \left( \sum_{j=1}^{n} a_{ij} q_j \right) p_i = \sum_{i=1}^{m} \sum_{j=1}^{n} p_i a_{ij} q_j = p^T A q. \tag{2.2.1}
\]

If player II’s strategy is known, player I would attempt to maximize the payoff by choosing a pure strategy or a mixed strategy:

\[
\max_{1 \leq i \leq m} \sum_{j=1}^{n} a_{ij} q_j = \max_{p \in X^*} p^T A q. \tag{2.2.2}
\]

The equation 2.2.2 holds because the left side is less than or equal to the right side since a pure strategy is a special mixed strategy with probability one in it, and the right side is also less than or equal to the left side since \( p^T A q \) is an average of \( \sum_{j=1}^{n} a_{ij} q_j \) so it must be less than or equals to the largest one. Any \( p \in X^* \) including pure strategies that gives the maximum average payoff is called a best response or a Bayes strategy against \( q \). Similarly, knowing that
player I is going to use a particular strategy \( p \in X^* \), player II would attempt to minimize his payoff to player I by choosing a pure strategy or a mixed strategy.

\[
\min_{1 \leq j \leq n} \sum_{i=1}^{m} p_i a_{ij} = \min_{q \in Y^*} p^T A q.
\]  

Any \( q \in Y \) including pure strategies that gives the minimum payoff is called a best response or a Bayes strategy against \( p \).

The minimum of equation 2.2.2 over all \( q \in Y^* \) is known as the upper value of the game \((X, Y, A)\) denoted by \( V \):

\[
V = \min_{q \in Y^*} \max_{1 \leq i \leq m} \sum_{j=1}^{n} a_{ij} q_j = \min_{q \in Y^*} \max_{p \in X^*} p^T A q.
\]  

There always exists \( q \in Y^* \) in a finite game \((X, Y, A)\) that achieves the minimum in (2.2.4) since \( \max_{1 \leq i \leq m} \sum_{j=1}^{n} a_{ij} q_j \) or \( \max_{p \in X^*} p^T A q \) is a continuous function of \( q \), and \( Y^* \) is a closed bounded set. Any \( q \in Y^* \) that achieves the minimum in (2.2.4) is called a mini-max strategy for player II. Similarly, the lower value of the game is the maximum of (2.2.3) over all \( p \in X^* \) denoted by \( V \):

\[
V = \max_{p \in X^*} \min_{1 \leq j \leq n} \sum_{i=1}^{m} p_i a_{ij} = \max_{p \in X^*} \min_{q \in Y^*} p^T A q.
\]  

Any \( p \in X^* \) that achieves the maximum in equation 2.2.5 is called a mini-max strategy for player I, and it always exists in a finite game \((X, Y, A)\).

**Lemma 2.** In a finite game, both players have mini-max strategies.
Lemma 3. The lower value is less than or equal to the upper value,

\[ V \leq \bar{V}. \]

In general, this inequality holds for any real valued function, \( f : X^* \times Y^* \to \mathbb{R} \):

\[
\max_{x \in X^*} \min_{y \in Y^*} f(x, y) \leq \min_{y \in Y^*} \max_{x \in X^*} f(x, y).
\]

If \( V = \bar{V} \), we say that the value of the game exists denoted by \( V \) and \( V \) equals to both \( V \) and \( \bar{V} \). If the value of the game exists, the mini-max strategies each are the optimal strategy for the player I or the player II. We can reinterpret the mini-max theorem by simply saying that every finite game has a value, and both players have mini-max strategies.

Lemma 4. Suppose there are two payoff matrices \( A = (a_{ij}) \) and \( A' = (a'_{ij}) \), and \( a'_{ij} = ca_{ij} + b \), with \( c > 0 \), then the mini-max strategies for I and II of a game with \( A \) and a game with \( A' \) are the same. Also, if \( V \) and \( V' \) are the values of the two games respectively, then \( V' = cV + b \).

Linear programming provides a simple algorithm for solving finite games. Linear programming (Chvatal, 1983) is a method for the optimization of a linear objective function of some
variables subject to linear constraints of these variables. Solving the game problem from player
I's point of view becomes choosing $p_1, \cdots, p_m$ to

$$\text{maximize } \min_{1 \leq j \leq n} \sum_{i=1}^{m} p_i a_{ij}$$

subject to

$$p_1 + \cdots + p_m = 1$$

$$p_i \geq 0 \ \forall i \in \{1, \cdots, m\}.$$  

But this is not a linear program because the objective function is not a linear function of the
$p$’s.

However, we can construct an equivalent linear programming problem by adding one new
variable $v$. We attempt to maximize $v$ and restrict it to satisfy an inequality constraint that
$v \leq \min_{1 \leq j \leq n} \sum_{i=1}^{m} p_i a_{ij}$. So solving the game problem becomes choosing $v$ and $p_1, \cdots, p_m$ to

$$\text{maximize } v$$

subject to

$$v \leq \sum_{i=1}^{m} p_i a_{i1}, \cdots, v \leq \sum_{i=1}^{m} p_i a_{in}$$

$$p_1 + \cdots + p_m = 1$$

$$p_i \geq 0 \ \forall i \in \{1, \cdots, m\}.$$
From player II’s point of view, the problem becomes choosing $w$ and $q_1, \cdots, q_n$ to

\[
\begin{align*}
\text{minimize } w \\
\text{subject to } \quad w &\geq \sum_{j=1}^{n} a_{1j}q_j, \cdots, w \geq \sum_{j=1}^{n} a_{mj}q_j \\
q_1 + \cdots + q_n &\leq 1 \\
q_j &\geq 0 \quad \forall j \in \{1, \cdots, n\}.
\end{align*}
\]

The theory of duality in linear programming indicates that these two programs are dual programs, and the famous Duality Theorem (Boyd and Vandenberghe, 2004) shows that the dual programs gives the same value that it also implies the minimax Theorem. Knowing that the value of the game is positive, we can transform the linear programming into another one for computation benefits. Suppose $v > 0$ and let $x_i = p_i/v$. We can replace the constraint $p_1 + \cdots + p_m = 1$ with $x_1 + \cdots + x_m = 1/v$. Maximizing $v$ is equivalent to minimizing $1/v$ that is minimizing $x_1 + \cdots + x_m$. The problem becomes choosing $x_1, \cdots, x_m$ to

\[
\begin{align*}
\text{minimize } x_1 + \cdots + x_m \\
\text{subject to } \quad 1 &\leq \sum_{i=1}^{m} x_ia_{i1}, \cdots, 1 \leq \sum_{i=1}^{m} x_ia_{in} \\
x_i &\geq 0 \quad \forall i \in \{1, \cdots, m\}.
\end{align*}
\]
Note that the strategy for player I is given by \( p_i = vx_i \) for \( i \in \{1, \cdots, m\} \).

2.3 Markov Decision Process (MDP)

To mathematically investigate process prediction problems, we need some model assumptions. The Markov decision process (MDP) is a well studied and one of the most widely used process models in many disciplines, including optimal control, artificial intelligence, economics and psychology. Richard Bellman’s work \( \text{[Bellman, 1957]} \) \( \text{[Bellman and Kalaba, 1965]} \) makes significant contribution to MDP model. Readers can refer to \( \text{[Puterman, 2014]} \) \( \text{[Bertsekas et al., 1995]} \) for a rigorous treatment of MDP model.

Definition 2.10. A Markov decision process \( \text{[Bellman, 1957]} \) is defined as a tuple \( \langle S, A, T, R \rangle \) where

- \( S \) : a set of states,
- \( A \) : a set of actions,
- \( T : S \times A \times S \rightarrow \mathbb{R} \) is the state transition dynamics that
  \[
  T(s, a, s') = p(s_{t+1} = s'|s_t = s, a_t = a),
  \]
- \( r : S \times A \rightarrow \mathbb{R} \) is the reward function that an agent would receive
  for being in state \( s \) and executing action \( a \).

This is a very basic definition of MDP model. It assumes the first-order Markov property in the definition of state transition dynamics and reward function that the reward and the next state only depend on the current state and action rather than the entire history of states and
actions taken. The Markov independence property greatly reduces the complexity in modeling and solving process prediction problems. Throughout the Markov Decision Process section 2.3, we also assume both $S$ and $A$ are finite sets, and the MDP is a discrete time model.

A more general definition of MDP model may assume both state transition function and reward function or one of them is time dependent having time index in notation, $p_t(s'|s,a)$ or $r_t(s,a)$. When any of them is time dependent, we say it is non-stationary or time variant, otherwise it is stationary or time invariant. The reward may also be a random variable with the conditional probability distribution $p(r|s,a)$ given. In this case, $r(s,a)$ may be used to define the conditional expected reward:

$$r(s,a) = \mathbb{E}[R_t|s_t = s, a_t = a] = \sum_{r \in R} rp(r|s,a). \tag{2.3.1}$$

In some cases, the probability of next state and reward conditions on current state and action is given that $p(s',r|s,a) = p(s_{t+1} = s', r_{t+1} = r|s_t = s, a_t = a)$. This completely specifies the MDP model, since the state transition dynamics and the conditional reward probability each can be obtained by marginalizing out $s_{t+1}$ or $r_{t+1}$. When reward function also depends on the next state, the expected conditional reward can be defined as

$$r(s_{t+1},s_t,a_t) = \mathbb{E}[R_t|s_{t+1} = s', s_t = s, a_t = a] = \frac{\sum_{r \in R} rp(s',r|s,a)}{p(s'|s,a)}.$$
The goal of MDP problems is to obtain a policy so as to maximize expected long term reward. In general, a policy conventionally assumes having first-order Markov property:

\[ \pi : S \times A \rightarrow \mathbb{R} \quad \pi(a|s) = p(a_t = a|s_t = s). \]

After all, the state defined in the MDP model means sufficient information for making an action. In addition, often the optimal policy used in most learning methods is deterministic. If an action maximizes the long-term expected reward starting from a state, there is no reason to consider other actions even though they may lead to the same maximum long-term expected reward. A deterministic policy is simply a mapping from states to actions:

\[ \pi : S \rightarrow A \quad \pi(s) = a. \]

Interpreting the meaning of long term leads to two variations of the MDP model. The Finite-Horizon MDP model assumes the interaction process ends in a finite number of time steps, and the goal is then to maximize the expected accumulated reward up to a finite time \(T\):

\[ \mathbb{E} \left[ \sum_{t=1}^{T} R_t \right]. \quad (2.3.2) \]
The Infinite-Horizon MDP model assumes an infinite number of running interactions, and the infinitely accumulated expected reward the agent aims to maximize is discounted according to a discount factor $\gamma \in (0, 1]$:

$$\mathbb{E} \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right].$$  \hspace{1cm} (2.3.3)

Adding the discount factor ensures that the infinite summation of expected rewards will be finite. With a finite horizon, an optimal policy could change in a given state over time. In general, the non-stationary setting of state transition dynamics, reward and policy is preferred in Finite-Horizon MDP models. However, with infinite horizon, there is no reason for an optimal policy that behaves differently in the same state at different time steps. State transition dynamics, reward and policy are often assumed to be stationary in Infinite-Horizon MDP model.

We introduce more detail about the Infinite-Horizon MDP model in section 2.3.1 based on the survey paper (Kaelbling et al., 1996) and the textbooks (Russell, 2009; Sutton and Barto, 1998; Mohri et al., 2012) and Finite-Horizon MDP in section 2.3.2 based on the textbook (Puterman, 2014). Finally, we give a brief overview of partially observable MDP model in section 2.3.3 where problems with complete perception of states is impossible based on the survey paper (Kaelbling et al., 1998) and the textbook (Russell, 2009).

2.3.1 Infinite-Horizon MDP

Besides the basic MDP model defined in definition 2.10 and infinitely accumulated discounted expected reward (2.3.3), Infinite-Horizon MDP in this section also assumes stationary policy $\pi(s)$ or $\pi(a|s)$.
**Definition 2.11.** The value of a state $s$ under a policy $\pi$ is defined as the expected reward returned starting from being in state $s$ and following $\pi$ thereafter:

$$v_\pi(s) = \mathbb{E} \left[ \sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k-1} \mid s_t = s \right]. \tag{2.3.4}$$

**Proposition 2.1.** The value of a state $s$ under a policy $\pi$ equals to the expected value of next state plus expected immediate reward:

$$\forall s \in S, \quad v_\pi(s) = \sum_a \pi(a|s) \left( r(s,a) + \gamma \sum_{s'} p(s_{t+1} = s' | s_t = s, a_t = a) v_\pi(s') \right). \tag{2.3.5}$$

Equations above (Equation (2.3.5)) are known as Bellman equations (Bellman, 1957) for policy $\pi$. If the state transition dynamics and reward function are completely known, the Bellman equations are a system of $|S|$ linear equations in $|S|$ unknown values $v_\pi(s)$. We can rewrite the Bellman equations to be

$$V = R + \gamma PV,$$

and it can be shown that $(I - \gamma P)$ is invertible, so the unique solution takes the following form:

$$V = (I - \gamma P)^{-1} R.$$

By the definition of value function, for an agent being in a state $s$, a policy $\pi$ is better than or equal to $\pi'$ if $v_\pi(s) \geq v_{\pi'}(s)$. In the Infinite-Horizon MDP setting, there is always at least
one policy that is better than or equal to all other policies for all $s \in S$. This gives the formal
definition of optimal policy.

**Definition 2.12.** A policy $\pi^*$ is optimal, if it gives maximal state value $v^*(s)$ for all $s \in S$.

By the definition above, for any $s \in S$, $v^*(s) = \max_\pi v_\pi(s)$.

**Definition 2.13.** The value of a state $s$ and action $a$ under a policy $\pi$ is defined as the expected
reward returned starting from being in state $s$ and taking action $a$ and following $\pi$ thereafter:

$$q_\pi(s,a) = E \left[ \sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k-1} \bigg| s_t = s, a_t = a \right].$$

(2.3.6)

Similar to proposition 2.1

$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'_{t+1} = s' | s_t = s, a_t = a) v_\pi(s').$$

(2.3.7)

The maximal state action value $q^*(s,a)$ is defined as

$$q^*(s,a) = \max_\pi q_\pi(s,a).$$

(2.3.8)

It is easy to see that

$$q^*(s,a) = r(s,a) + \gamma \sum_{s'} p(s'_{t+1} = s' | s_t = s, a_t = a) v^*(s').$$

(2.3.9)
We can also notice that

\[ v^*(s) = \max_a q^*(s, a). \]  

(2.3.10)

By definition 2.12 there exists a deterministic optimal policy such that

\[ \forall s \in S, \quad \pi^*(s) = \arg \max_a q^*(s, a). \]

Knowing \( q^*(s, a) \) is sufficient to obtain a deterministic optimal policy \( \pi^*(s) \), while we can obtain \( q^*(s, a) \) by knowing \( v^*(s) \). Substituting \( q^*(s, a) \) in equation (2.3.10) by equation (2.3.9) gives the recursively defined equations of \( v^*(s) \) known as Bellman optimal equations:

\[ \forall s \in S, \quad v^*(s) = \max_a \left( r(s, a) + \gamma \sum_{s'} p(s_{t+1} = s'|s_t = s, a_t = a)v^*(s') \right). \]  

(2.3.11)

Because of the max operator, the Bellman optimal equations are nonlinear. One may consider to solve the equations directly by using any of the methods for solving the systems of nonlinear equations, but often this does not work in practice due to computation complexity. Obtaining the optimal value function means exhaustively searching the value function space requiring high-demand of computation time and resource.

Solving the Bellman optimal equation via dynamic programming methods is often preferred in practice. We introduce two dynamic programming methods that are the policy iteration and the value iteration methods. The policy iteration method consists of policy evaluation and policy improvement steps. The policy evaluation step solves the systems of Bellman equations
to obtain the value function $v(s)$ for any given policy $\pi$. The policy improvement step updates the policy by making it greedy with respect to the state action value function that $\pi(s) = \arg \max_a q(s, a)$. The basic procedure of the policy iteration method starts with an arbitrary policy. It then performs policy evaluation and policy improvement iteratively until a stop condition is matched. A typical stop condition could be if the maximal absolute difference of successive value functions of a state is sufficiently small. Solving the Bellman equations directly in the policy evaluation step may also be computationally inefficient for reinforcement learning problems with a large state space. Iterative method that obtains an approximate value function via an update rule may be more suitable. This method is called iterative policy evaluation, and it starts with an initial value function $v(s)$ that it improves by using the Bellman equation 2.3.5 with a deterministic policy $\pi(s)$ as an update rule that

$$\forall s \in S, \quad v(s) = r(s, \pi(s)) + \gamma \sum_{s'} p(s_{t+1} = s' | s_t = s, a_t = \pi(s)) v(s')$$

until a stop condition is matched.

The value iteration method instead updates the value function iteratively by making it greedy with respect to available actions. The basic procedure of the value iteration method starts with an arbitrary value function $v(s)$, and improves the value function by using the Bellman optimal equation 2.3.11 as an update rule:

$$\forall s \in S, \quad v(s) = \max_a \left( r(s, a) + \gamma \sum_{s'} p(s_{t+1} = s' | s_t = s, a_t = a) v(s') \right)$$

(2.3.12)
until the stop condition of value improvement is matched.

There are mathematical theorems showing that both the policy iteration method and the value iteration method guarantee the convergence of the value sequence to the optimal one $v^*$ which then leads to the optimal policy $\pi^*$ (Sutton and Barto, 1998). The complete algorithms of the two methods are given in algorithm 2.1 and 2.2 respectively.

**Algorithm 2.1 Policy iteration**

**Input:** Arbitrary $V(s)$ and $\pi(s)$ for all $s \in S$ and a sufficient small number $\epsilon$

**Output:** $\pi$

```
\text{repeat}
\quad \pi' = \pi
\text{repeat}
\quad \Delta \leftarrow 0
\quad \text{for each } s \in S \text{ do}
\quad \quad v \leftarrow v(s)
\quad \quad v(s) = r(s, \pi'(s)) + \gamma \sum_{s'} p(s'_{t+1} = s' | s_t = s, a_t = \pi'(s)) v(s')
\quad \quad \Delta \leftarrow \max(\Delta, |v - v(s)|)
\quad \text{end for}
\text{until } \Delta \leq \epsilon
\text{for each } s \in S \text{ do}
\quad \pi(s) = \arg \max_a (r(s, a) + \gamma \sum_{s'} p(s'_{t+1} = s' | s_t = s, a_t = a) v(s'))
\text{end for}
\text{until } \pi = \pi'
\text{return } \pi
```
Algorithm 2.2 Value iteration

Input: Arbitrary $V(s)$ for all $s \in S$ and a sufficient small number $\epsilon$
Output: $\pi$

repeat
  $\Delta \leftarrow 0$
  for each $s \in S$ do
    $v \leftarrow v(s)$
    $v(s) = \max_a \left( r(s, a) + \gamma \sum_{s'} p(s_{t+1} = s' | s_t = s, a_t = a) v(s') \right)$
    $\Delta \leftarrow \max(\Delta, |v - v(s)|)$
  end for
until $\Delta \leq \epsilon$
for each $s \in S$ do
  $\pi(s) = \arg \max_a \left( r(s, a) + \gamma \sum_{s'} p(s_{t+1} = s' | s_t = s, a_t = a) v(s') \right)$
end for
return $\pi$

2.3.2 Finite-Horizon MDP

As for Finite-Horizon MDPs, besides the basic definition of the MDP, we assume non-stationary state transition dynamics $T_t$ and policy $\pi$. Given a non-stationary policy $\pi = (\pi_1, \cdots, \pi_T)$, we use $J_\pi$ to denote the expected total reward up to the finite time $T$:

$$J_\pi = \mathbb{E} \left[ \sum_{t=1}^{T} R_t \right].$$

(2.3.13)

If we assume the initial state $s_1 = s$ is given, the expected total reward is

$$J^s_\pi = \mathbb{E} \left[ \sum_{t=1}^{T} R_t | s_1 = s \right].$$

(2.3.14)

Note that in a Finite-Horizon MDP, the value function is also non-stationary.
**Definition 2.14.** The value of a state $s$ at a given time $t$ under a policy $\pi$ is defined as the expected reward returned starting from being in state $s$ at time $t$ and following $\pi$ thereafter until a final time $T$:

$$v_\pi^t(s) = \mathbb{E} \left[ \sum_{k=t}^{T} R_k \mid s_t = s \right].$$  \hspace{1cm} (2.3.15)

**Proposition 2.2.** The value of a state $s$ at a given time $t$ under a policy $\pi$ equals to the expected value of next state at time $t+1$ plus expected immediate reward:

$$\forall s \in S, \quad v_\pi^t(s) = \sum_a \pi_t(a|s) \left( r_t(s,a) + \sum_{s'} p_t(s'_{t+1} = s' \mid s_t = s, a_t = a) v_\pi^{t+1}(s') \right).$$  \hspace{1cm} (2.3.16)

**Definition 2.15.** A policy is optimal, denoted by $\pi^*$, if it gives maximal state value $v_\pi^*(s)$ for all $s \in S$ and $t = 1, \cdots, T$.

We can obtain the optimal value sequence $v_1^*, \cdots, v_T^*$ and the optimal policy $\pi^* = (\pi_1^*, \cdots, \pi_T^*)$ via backward recursion equations:

$$v_t^*(s_t) = \max_{a_t \in A_t} \left\{ r_t(s_t, a_t) + \sum_{s'_{t+1} \in S_{t+1}} p_t(s'_{t+1} \mid s_t, a_t) V_{t+1}^*(s'_{t+1}) \right\},$$  \hspace{1cm} (2.3.17)

$$\pi_t^*(s_t) = \arg \max_{a_t \in A_t} \left\{ r_t(s_t, a_t) + \sum_{s'_{t+1} \in S_{t+1}} p_t(s'_{t+1} \mid s_t, a_t) V_{t+1}^*(s'_{t+1}) \right\}.$$  \hspace{1cm} (2.3.18)

A typical dynamic programming algorithm for solving Finite-Horizon MDP problems starts with the setting $t = T$ and $v_T^*(s_T) = r_T(s_T)$ for all $s_T \in S$, and then recursively substitutes $t - 1$ for $t$ to compute $V_t^*(s_t)$ and obtain $\pi_t^*(s_t)$ for each $s_t \in S$ by equation 2.3.17 and 2.3.18 respectively until $t = 1$. Note that $\pi^*$ maximizes $v_1^*(s)$ simultaneously for every initial state.
\[ s_1 = s \text{ so as } J^*_s = s. \text{ If the distribution of initial state } p_1(s) \text{ is given, since } \\
J^* = \sum_s p_1(s) J^*_s, \] 
(2.3.19)

it’s easy to see that \( \pi^* \) maximizes the expected total reward \( J^* \).

2.3.3 Partially Observable MDP (POMDP)

We have seen the formulation and dynamic programming solutions of MDP models in previous sections. It assumes an agent can sense all necessary information for making an action. In other words, a fully observable environment is assumed. For example, in playing Chess, players know the locations of chess pieces on the board, which is sufficient for players to make a decision at each time step where a MDP model is a good fit. Unfortunately, it is not true for many other process prediction problems. For instance, in playing a Poker game, players can only see their own cards but not the ones others hold since the process environment is partially observable. Players play a card based on their belief of how the cards others hold are distributed from not only seeing his or her cards on hand but also all cards that all players have taken out. The belief keeps being updated as the game continues. A process prediction model that fits the situation corresponding to this case is known as a partially observable MDP (POMDP).

Definition 2.16. A partially observable Markov decision process (POMDP) can be described as a tuple \( \langle \mathcal{S}, \mathcal{A}, T, R, \Omega, O \rangle \) where

- \( \langle \mathcal{S}, \mathcal{A}, T, R \rangle \) is the same to the definition 2.10 of MDP,
- \( \Omega \) is a finite set of observations,
\[ O : S \times A \times \Omega \rightarrow \mathbb{R}, \text{ is the observation dynamics that} \]

\[ O(s', a, o') = p(o' \mid s', a). \]

In this section, we assume the POMDP model is infinite-horizon, and has a stationary state transition dynamics, observation dynamics and a stationary policy. The goal is to find a policy so as to maximize the infinite accumulated discounted expected reward. In a partially observable environment, exact states values are unknown to the agent. Directly solving the POMDP problem is challenging. An efficient way is to maintain a recursively defined belief state \( b \) that is the probability distribution of real states summarizing the information of previous actions and observations. Let \( b(s) \) denotes the probability of state \( s \) in belief state \( b \), we have \( 0 \leq b(s) \leq 1 \) and \( \sum_s b(s) = 1 \). Given a belief state \( b \), an action \( a \) that the agent executes following an immediate observation \( o' \), the probability distribution of states in the new belief state \( b' \) can be defined as

\[ b'(s') = p(s' \mid b') = p(s' \mid b, a, o') \quad (2.3.20) \]

where

\[ p(s' \mid b, a, o') \propto p(o', s' \mid b, a) = O(s', a, o') \sum_s T(s, a, s') b(s). \quad (2.3.21) \]

To simplify the description of problem, we define the state estimation function \( SE \) such that \( b' = SE(b, a, o) \) if \( b'(s') \) matches the equation 2.3.20 given the belief state \( b \), action \( a \) and observation \( o \). The main motivation of defining belief states is that an action will only depends
on the current belief state getting rid of the sequence of previous actions and observations that
the policy is a mapping from belief states to actions:

\[ \pi : B \rightarrow A \quad \pi(b) = a. \]

Formally, we can reduce the POMDP model to so called Belief MDP model [Kaelbling et al., 1998].

**Definition 2.17.** A belief Markov decision process (Belief MDP) is defined as a tuple

\[ < B, A, T^b, R^b > \]

where

- \( B \) is a set of belief states,
- \( A \) is a set of actions,
- \( T^b : B \times A \times B \rightarrow \mathbb{R} \) is the stationary transition probability of belief states that

\[ T^b(b, a, b') = p(b'|b, a) = \sum_{o'} p(b'|b, a, o') p(o'|b, a) \] and

\[ p(b'|b, a, o') = \begin{cases} 1 & \text{if } SE(b, a, o') = b' \\ 0 & \text{otherwise} \end{cases} \quad (2.3.22) \]

- \( r^b : B \times A \rightarrow \mathbb{R} \) is the expected reward that an agent would receive for being in belief state

\[ b \] and executing action \( a \) that

\[ r^b(b, a) = \sum_s b(s) r(s, a). \]
The definition of value function, Bellman equation and policy of Belief MDP is similar to Infinite-Horizon MDP introduced in section 2.3.1. The Bellman optimal equation for the belief states is defined as

$$v^*(b) = \max_a \left( r^b(b, a) + \gamma \sum_{b'} p(b'|b, a) v^*(b') \right).$$  \hspace{1cm} (2.3.23)

Unfortunately, we can not directly apply the dynamic programming methods discussed in section 2.3.1 to solve Belief-MDP problems. Note that the belief state is a probability distribution over real states. This distribution corresponds to a continuous state space leading to infinitely many candidate beliefs. A number of methods based on value iteration or policy iteration have been proposed. However, finding optimal policy for general POMDP problems is still very challenging. Approximate methods are often preferred in practice. Readers can refer to (Sondik, 1971; Cassandra et al., 1994; Kaelbling et al., 1998; Pineau et al., 2003) for more details of solving general POMDP problems.

2.4 Reinforcement Learning

Reinforcement learning (Kaelbling et al., 1996; Sutton and Barto, 1998) is a computation approach which aims to learn a mapping from states to actions so as to maximize reward over an interaction process. The learner is called the agent, and the things it interacts with that cannot be changed arbitrarily by the agent is known as the environment.

Reinforcement learning is an important and active research topic in artificial intelligence, but it also intersects with other research communities (Sutton and Barto, 1998; Cross, 1973).
including economics, control, psychology, neuroscience and cognitive science with substantial benefits going both ways. In some sense, earlier work in the psychology concerning learning by trial and error (Klopf, 1982), optimal control using value function and dynamic programming (Bellman and Kalaba, 1965) and temporal-difference learning (Barto and Sutton, 1982) produced the field of modern reinforcement learning. There have been many successful applications of reinforcement learning in various decision making problems such as learning to play Backgammon, box-pushing robot, elevator dispatching, and job-shop scheduling (Tesauro, 1995; Mahadevan and Connell, 1992; Crites and Barto, 1996; Zhang and Dietterich, 1995).

Reinforcement learning is different from supervised learning which learns a mapping from features as input to labels or numerical values as output from a labeled training set provided by knowledgeable experts. Reinforcement learning is also different from unsupervised learning which attempts to find hidden patterns in collected unlabeled data.

The conventional setting of reinforcement learning assumes problems with a discrete number of time steps, but the general principle of reinforcement learning also works for continuous setting (Doya, 2000). Reinforcement learning can also deal with problems with very large states, or even infinite (Tesauro, 1995).

A reinforcement learning problem typically consists of four main elements (Sutton and Barto, 1998). A policy defines the behavior of an agent that it is a mapping from states to actions. A reward function defines an immediate payoff that an agent can receive after executing an action from a state. A value function of a state specifies expected long-term reward an agent can receive via a given policy, starting from the state. A model of an environment specifies
how the interaction process of an agent and its environment involves. Most of reinforcement
learning techniques investigate problems in the MDP (Definition 2.10) setting where the model
is mainly specified by the state transition dynamics. If the state transition dynamics is given,
the problem becomes a planning problem for which the agent can obtain an optimal policy by
just solving a MDP problem via dynamic programming methods.

However, reinforcement learning problems assuming the model is unknown better match
many real agent behavior learning problems. The agent needs to interact with its environment
to adjust its behavior via information it continuously receives mainly consisting of reward and
observations. There have been two main approaches to solve reinforcement learning problems.
One is to estimate the model first and then use the model to obtain the policy, known as
the model-based approach. The other one is to learn the policy directly without learning the
model, known as the model-free approach. There is no convincing argument that indicates
which of the two approaches is better (Kaelbling et al., 1996). The Model-free approach is
much more popular since it avoids model estimation, but it requires a great deal of interaction
experience to achieve good performance. So the model-based approach might be preferred
if exploration is considered to be costly. Moreover, it is useful to combine model-based and
model-free methods to formulate a more general learning model yielding both performance
effectiveness and computation efficiency (Sutton, 1991).

The agent, in one hand, has to exploit what it already knows to accumulate reward. In
the other hand, the agent has to explore unknown part of the environment to make better
actions in future. The trade-off between exploitation and exploration is one of the challenges
in reinforcement learning. Though there is no standard method that optimally balance the exploitation and exploration for general reinforcement learning problems, a number of methods proposed for the trade-off [Berry and Fristedt, 1985, Thrun, 1992] can lead to promising learning performance for many reinforcement learning problems in practice. For example, the ε – greedy policy takes the optimal action that maximizes the expected long-term reward most of the time, but with probability ε instead chooses one of the other actions of random. So, in long term, it ensure that all actions are selected infinitely often.

Modern reinforcement learning methods are developed to balance the performance and computation complexity, taking into account the benefits of exploration, while coping with high dimensional and large-scale states space. But the essential idea is still iteratively improving a value function of state or value function of state and action via appropriate updating rules so as to obtain an optimal or near optimal policy. Algorithm 2.3 gives a model-free learning method known as Q-learning considered as one of the most important breakthroughs in reinforcement learning. The update rule of Q-learning is

\[ q(s,a) = q(s,a) + \alpha \left( r + \gamma \max_{a'} q(s',a') - q(s,a) \right) \]  

(2.4.1)

which essentially is the empirical version of Bellman optimal equation 2.3.9 where \(<s,a,r,s'>\) is the empirical interaction tuple. If each action is executed an infinite number of times in each state and α is decayed appropriately, q has been shown to converge to the optimal one, q*, with probability 1.
Readers can refer to Sutton and Barto’s textbook (Sutton and Barto, 1998) for the details and a broader view of reinforcement learning techniques. Additionally, Bertsekas’s textbook (Bertsekas et al., 1995) provides a more mathematically rigorous treatment of the essential elements of reinforcement learning.

**Algorithm 2.3 Q-learning**

<table>
<thead>
<tr>
<th>Input: Arbitrary $q(s,a)$ $\forall s \in S, a \in A$</th>
<th>Output: $\pi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial state $s$</td>
<td></td>
</tr>
<tr>
<td>repeat</td>
<td></td>
</tr>
<tr>
<td>$a \leftarrow \text{argmax}_a q(s,a)$</td>
<td></td>
</tr>
<tr>
<td>execute $a$ and observe $r$ and $s'$</td>
<td></td>
</tr>
<tr>
<td>$q(s,a) = q(s,a) + \alpha (r + \gamma \max_{a'} q(s',a') - q(s,a))$</td>
<td></td>
</tr>
<tr>
<td>$s \leftarrow s'$</td>
<td></td>
</tr>
<tr>
<td>until interaction end</td>
<td></td>
</tr>
<tr>
<td>for each $s \in S$</td>
<td></td>
</tr>
<tr>
<td>$\pi(s) = \text{argmax}_a q(s,a)$</td>
<td></td>
</tr>
<tr>
<td>end for</td>
<td></td>
</tr>
<tr>
<td>return $\pi$</td>
<td></td>
</tr>
</tbody>
</table>

### 2.5 Optimal Control

Optimal control (Stengel, 2012) aims to design a controller so as to maximize some measure of performance or to minimize a cost function over time. For instance, fuel used might be a cost function that an aircraft is designed to minimize. Optimal control is a large topic that it is too wide to discuss here. In this section, we introduce two important optimal control problems that
are related to our work. Both of them assume a quadratic cost function and linearly defined state transition dynamics with Gaussian noise. The first one is the linear quadratic regulator (LQR) used to model a linear control system with a fully observable environment, and the other one is the linear quadratic Gaussian system (LQG) which assumes a partially observable environment. More specifically, we investigate discrete-time, finite-horizon, stochastic LQR and LQG problems. With quadratic cost function and linear process assumption, both problems provide closed-form solutions from solving the Bellman optimal equations.

2.5.1 Linear-Quadratic Regulator (LQR)

Definition 2.18. A Finite-Horizon, discrete-time, stochastic linear quadratic regulator is defined by the following linear system equations:

\[ x_{t+1} = Ax_t + Bu_t + \varepsilon \quad t = 1, \cdots, T, \]  

where \( x_t \) is the state of the system, \( u_t \) is the control, \( A \) and \( B \) matrices define the linear relationship of the system, \( \varepsilon \) is the process noise with mean \( \mathbb{E}[\varepsilon] = 0 \) and variance \( \text{Var}(\varepsilon) = \Sigma \) and a quadratic cost function:

\[ J = \mathbb{E} \left[ \sum_{t=1}^{T} (X_t^T Q X_t + U_t^T R U_t) + U_{T+1}^T Q_f U_{T+1} \right] \]  

where \( Q \succeq 0, Q_f \succeq 0 \) are positive-semidefinite matrices, and \( R \succ 0 \) is positive-definite matrix.

Solving the LQR problem requires finding a policy that minimizes the cost function \( J \). It’s easy to see that LQR is a special case of a Finite-Horizon MDP discussed in section 2.3.2.
Instead of an analytic solution for general Finite-Horizon MDPs, solving the Bellman optimal equations of LQR leads to a closed-form solution. More specifically, the optimal policy \((u_1, \cdots, u_T)\) is revealed by a linear state feedback function that is non-stationary and deterministic:

\[
    u_t = -L_t x_t
\]

where

\[
    L_t = (B^T F_{t+1} B + R)^{-1} B^T F_{t+1} A
\]

with

\[
    F_t = \begin{cases} 
        A^T F_{t+1} A - A^T F_{t+1} B (B^T F_{t+1} B + R)^{-1} B^T F_{t+1} A + Q & t \leq T, \\
        Q_f & t = T + 1.
    \end{cases}
\]

### 2.5.2 Linear-Quadratic-Gaussian System (LQG)

**Definition 2.19.** A **Finite-Horizon discrete-time stochastic linear quadratic Gaussian system** is defined by the following linear system equations:

\[
    x_{t+1} = Ax_t + Bu_t + \varepsilon_s \quad t = 1, \cdots, T, \quad (2.5.6)
\]

\[
    z_t = Cx_t + \varepsilon_o \quad t = 1, \cdots, T, \quad (2.5.7)
\]

where \(x_t\) is the state of the system, \(u_t\) is the control, \(A\), \(B\) and \(C\) matrices define the linear relationship of the system, the initial state \(X_1 \sim N(\mu, \Sigma_s)\), the process noise \(\varepsilon_s \sim N(0, \Sigma_s)\) and
observation noise $\varepsilon_o \sim N(0, \Sigma_o)$ all follow Gaussian distribution and a quadratic cost function:

$$J = \mathbb{E} \left[ \sum_{t=1}^{T} (X_t^T Q X_t + U_t^T R U_t) + U_{T+1}^T Q_f U_{T+1} \right]$$

(2.5.8)

where $Q \succeq 0, Q_f \succeq 0$ are positive-semidefinite matrices, and $R \succ 0$ is positive-definite matrix.

Solving the LQG problem requires finding a policy that minimizes the cost function $J$. Note that LQG is a special case of a Finite-Horizon POMDP. Similar to a POMDP, an efficient way for solving the LQG problem is to maintain a quantity that is sufficient to make an action decision. For LQG, this sufficient quantity is the state estimate $\hat{x}_t$, which is the mean of the state conditioned on previous and current observations $\mathbb{E}[X_t|z_1, \cdots, z_t]$. Solving the Bellman optimal equations of LQG leads to a closed form solution. More specifically, the optimal policy $(u_1, \cdots, u_n)$ is revealed by a linear state feedback function that is non-stationary and deterministic:

$$u_t = L_t \hat{x}_t, \quad (2.5.9)$$

where $L_t$ is exactly the same as the one defined by equation 2.5.4 in which $F_t$ takes the form shown in the equation 2.5.5. The state estimate is obtained recursively:

$$\hat{x}_{t+1} = A\hat{x}_t + B u_t + K_{t+1}(z_{t+1} - C(A\hat{x}_t + B u_t)),$$  

(2.5.10)

with initial setting $\hat{x}_1 = \mathbb{E}[X_1]$. $K_t$ is called the Kalman gain:

$$K_t = P_t C^T (C P_t C^T + \Sigma_o)^{-1}, \quad (2.5.11)$$
where $P_t$ is determined by the following matrix Riccati difference equation that runs forward in time:

$$P_{t+1} = A(P_t - P_tC^T(CPC^T + \Sigma_o)^{-1}CP_t)A^T + \Sigma_s$$ (2.5.12)

with $P_1 = \Sigma_s$.

### 2.6 Covariate Shift

Most of machine learning methods (Vapnik and Vapnik, 1998; Bishop, 2006; Friedman et al., 2001) assume that the data used for training and testing follow the same probability distribution.

Given a loss function $L(x, y, f(x))$ which measures the discrepancy between the true output $y$ and predictor $f(x)$ on a particular input $x$, the expected test loss is expressed as

$$R_{te}(f(x)) = \mathbb{E}_{Xte, Yte}[L(X, Y, f(X))].$$ (2.6.1)

Then the optimal predictor is

$$f^*(x) = \arg \min_{f(x)} R_{te}(f(x)).$$ (2.6.2)

Let $f(x; \theta)$ be a predictor restricted to a parametric form, we say that a model in terms of the parametric form is correctly specified if there exists a parameter $\theta^*$ such that

$$f(x; \theta^*) = f^*(x).$$ (2.6.3)
Otherwise, the model is misspecified.

The optimal parameter for testing in terms of the parametric form is

$$\theta^*_{te} = \arg\min_{\theta} R^{te}(f(x; \theta)).$$

(2.6.4)

In practice, we can only learn $\theta$ from training data:

$$\hat{\theta}_{tr} = \arg\min_{\theta} \mathbb{E}_{X^{tr},Y^{tr}} [L(X,Y,f(X;\theta))] = \arg\min_{\theta} \left[ \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} L(x_i, y_i, f(x_i; \theta)) \right].$$

(2.6.5)

Accordingly, we define the optimal parameter for training to be $\theta^*_{tr}$. It turns out that $\hat{\theta}_{tr}$ is a consistent estimator of $\theta^*_{tr}$, meaning $\hat{\theta}_{tr}$ converges to $\theta^*_{tr}$ in probability even for misspecified models:

$$\lim_{N_{tr} \to \infty} \hat{\theta}_{tr} \xrightarrow{p} \theta^*_{tr}.$$  

(2.6.6)

Ideally, if $p_{tr}(x,y) = p_{te}(x,y)$, $\theta^*_{tr} = \theta^*_{te}$ then $\hat{\theta}_{tr}$ converges to $\theta^*_{te}$ in probability.

However, the assumption that training and testing data sharing the same probability distribution is often not true in many real-world applications such as behavior modeling, robot control, and brain signal analysis. For many process prediction problems, the process setting under which the data is collected, and the one to which the prediction model is deployed are often different. Thus, there is a strong need for developing prediction models that are adaptive to changing process settings. However, we need to assume the process settings of training and
testing share some common property. After all, we want to learn a prediction model from training data for testing.

Covariate shift (Sugiyama and Kawanabe, 2012) is the situation where the input distributions of training and testing are different, \( p_{tr}(x) \neq p_{te}(x) \), while the conditional distributions of output given the same input value, \( p(y|x) \), are the same. Under covariate shift, \( \hat{\theta}_{tr} \) is still a consistent estimator of \( \theta_{te}^* \) if the model is correctly specified, but it is no longer consistent if the model is misspecified (Sugiyama and Kawanabe, 2012).

### 2.6.1 Importance Reweighted Method

We first assume that the ratio of test to training input densities which is called importance is bounded, that is

\[
\frac{p_{te}(x)}{p_{tr}(x)} < \infty \text{ for all } x \in \mathcal{X}.
\]

More precisely, we require the importance ratio’s first moment to be finite (Cortes et al., 2010):

\[
\mathbb{E}_{p_{tr}(x)} \left[ \frac{p_{te}(X)}{p_{tr}(X)} \right] < \infty.
\]

Then

\[
\mathbb{E}_{X_{te}, Y_{te}} \left[ L(X, Y, f(X; \theta)) \right] = \int_X p_{te}(x) \int_{Y|X} p(y|X = x)L(x, y, f(x; \theta))
\]

\[
= \int_X p_{tr}(x) \frac{p_{te}(x)}{p_{tr}(x)} \int_{Y|X} p(y|X = x)L(x, y, f(x; \theta))
\]

\[
= \mathbb{E}_{X_{tr}, Y} \left[ \frac{p_{te}(X)}{p_{tr}(X)} L(X, Y, f(X; \theta)) \right].
\]
This is called the importance-weighted expectation \cite{Sugiyama2012} of the function over the training input $x^{tr}$. Hence

$$
\theta^*_te = \theta^*_triw = \arg \min_{\theta} \mathbb{E}_{X^{tr}, Y} \left[ \frac{p_{te}(X)}{p_{tr}(X)} L(X, Y, f(X; \theta)) \right].
$$

(2.6.7)

Let

$$
\hat{\theta}_{triw} = \arg \min_{\theta} \left[ \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \left( \frac{p_{te}(x_i)}{p_{tr}(x_i)} \right)^\gamma L(x_i, y_i, f(x_i; \theta)) \right],
$$

(2.6.8)

then $\hat{\theta}_{triw}$ is a consistent estimator of $\theta^*_te$ that

$$
\lim_{N_{tr} \to \infty} \hat{\theta}_{triw} \xrightarrow{p} \theta^*_triw = \theta^*_te.
$$

(2.6.9)

Intuitively, the importance-weighted model chooses informative training samples by considering the importance of each training sample. Training samples that have a distance from the test region are less informative. Thus the estimator $\hat{\theta}_{triw}$ may rely on very small amount of training examples making it an unstable estimator. Therefore importance-weighted model may not work well for small training samples. A slightly stabilized variant of importance-weighted model called adaptive importance-weighted model is achieved by slightly “flattening” the importance weight is preferable:

$$
\hat{\theta}_{\gamma,\triw} = \arg \min_{\theta} \left[ \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \left( \frac{p_{te}(x_i)}{p_{tr}(x_i)} \right)^\gamma L(x_i, y_i, f(x_i; \theta)) \right]
$$

(2.6.10)
where \(0 \leq \gamma \leq 1\) is called the flattening parameter providing the trade-off between stability and consistency of the model, known as the bias-variance trade-off. A good choice of \(\gamma\) roughly depends on the number of training data \(N_{tr}\). When \(N_{tr}\) is small, the model may be inclined to have high variance that is less stable, and hence small \(\gamma\) is appropriate. On the other hand, when \(N_{tr}\) is large, large \(\gamma\) is preferable leading to a smaller bias estimator that is more consistent.
CHAPTER 3

RELATED WORK

This chapter introduces related work on developing prediction models for process prediction problems. **Direct estimation** is a straightforward and simple strategy for solving process prediction problems using classical supervised learning methods. **Inverse reinforcement learning** (IRL), also known as **inverse optimal control** (IOC), attempts to recover a reward function from given training samples so as to model learning using reinforcement learning. Section 3.1 introduces the direct estimation approach. The details of inverse reinforcement learning are given in section 3.2.

### 3.1 Direct Estimation

Direct estimation, also known as behavior cloning, basically formulates a process prediction problem as a classical supervised learning problem. Classical supervised learning methods such as logistic regression, linear regression, support vector machines and neural networks have been widely used in earlier work on solving process prediction tasks. For example, Pomerleau (Pomerleau, 1989) built an artificial neural network to learn a prediction model to steer an autonomous vehicle driving at normal speed on public highways. Some other examples include Sammut, Kuniyoshi, Hayes and Demiris’s work (Sammut et al., 1992; Kuniyoshi et al., 1994; Hayes and Demiris, 1994). Many process prediction problems concern inferring positional distribution such as the position of a robot arm, an aerocraft drone, a human or of animals, a
continuous quantity. One of the most notable and widely used direct estimation approaches for
dealing with continuous quantity is known as linear regression.

3.1.1 Linear Regression

Assume we are given a list of sequences of states: $s_{1:T_N}$ where $T$ denotes the length of a
sequence, and $N$ is the number of sequences. Given current state $s_t$, we are concerned with what
the next state $s_{t+1}$ should be, which assumes the state transition dynamics is deterministic.
Without loss generality, in this section, we suppose $s_t$ is a one dimensional numerical value, and
let $x$ denotes $s_t$ and $y$ denotes $s_{t+1}$ being the input and output respectively. Linear regression
assumes a linear relationship between input variables $x$ and output variable $y$:

$$\hat{y}_{a,b}(x) = bx + a \quad (3.1.1)$$

that is parameterized by weights $a$ and $b$. A standard method for estimating these parameters
is to minimize the sum of squared residuals between estimator $\hat{y}_{a,b}(x)$ and the actual output
variable $y$ that is $s_{t+1}$ for $t = 1, \cdots, T_N - 1$. This is equivalent to an expected square loss over
the empirical distribution $\tilde{f}(x)\tilde{f}(y|x)$:

$$\arg\min_{a,b} \mathbb{E}_{\tilde{f}(x)\tilde{f}(y|x)} \left[ (Y - \hat{y}_{a,b}(X))^2 \right]. \quad (3.1.2)$$
This optimization (Equation [3.1.2]) is equivalent to maximizing the conditional log likelihood of sample data:

$$
\arg\max_{a,b,\sigma} \mathbb{E}_{f(x)} f(y|x) \left[ \log \hat{f}_{a,b,\sigma}(Y|X) \right] \tag{3.1.3}
$$

with $\hat{f}(y|x) \sim N(\hat{y}_{a,b}(x), \sigma^2)$ normally distributed with mean $\hat{y}_{a,b}(x)$ and variance $\sigma^2$ (Bishop, 2006). This corresponds to a zero-mean Gaussian residual error model with variance $\sigma^2$.

The linear regression model of equation [3.1.2] assumes the current state $s_t$ is sufficient to determine the next state $s_{t+1}$. We say it has a first-order Markov independence property. Sacrificing some computational efficiency, we may relax this property to $k$th-order Markov independence so as to improve the prediction accuracy:

$$
\hat{s}_{t+1} = b^T [s_t, s_{t-1}, \ldots, s_{t-k+1}] + \varepsilon \tag{3.1.4}
$$

with $\varepsilon \sim N(s, \sigma^2)$ a zero-mean Gaussian residual error model with variance $\sigma^2$.

Linear regression is a simple and widely adopted prediction model for many problems with continuous inputs and outputs. But it often has poor performance for process prediction problems even evaluating using the squared loss. It performs learning at a single or $k$ window size time frame ($k$th order Markov) without considering the process in its entirety. In addition, linear regression as a direct estimation approach is not adaptive or lacks of generalization capabilities for many process prediction tasks, especially the ones under decision-making settings. For instance, in a driving learning task, a classical supervised learning method that attempts
to mimic trajectories demonstrated by expert drivers from a small number of training trajectories would not work since the driving circumstance encountered each time is different. Linear regression often serves as a baseline to compete with when new process prediction methods are developed.

### 3.2 Inverse Reinforcement Learning

The early definition of inverse reinforcement learning (IRL) \cite{Russell1998} (also known as inverse optimal control (IOC)) attempts to model learning using reinforcement learning (Section 2.4) is formulated as a computation task.

**Definition 3.1.** Given

1. measurements of an agent’s behavior over time, in a variety of circumstances,
2. measurements of the sensory inputs to that agent;
3. a model of the physical environment (including the agent’s body);

determine the reward function that the agent is optimizing.

The basic principle \cite{Ng2000} of IRL (Definition 3.1) is mathematically formulated in a MDP (Definition 2.10) environment is to find a reward function $R^*$ such that

$$
\mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi^* \right] \geq \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi \right] \quad \forall \pi
$$

(3.2.1)

where $\pi^*$ is an expert policy. Since we typically only have expert demonstrations rather than the entire expert policy, the inequality above is not completely specified and cannot be solved.
Most IRL methods assume linear feature-based reward functions $R(s) = \omega^T \phi(s)$, where $\phi(s)$ is a feature function, so as to estimate accumulated feature expectation from expert demonstrations, and some of them are built on a linearity assumption:

$$E \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi \right] = \omega^T E \left[ \sum_{t=0}^{\infty} \gamma^t \phi(s_t) | \pi \right] = \omega^T \mu(\pi).$$ \hspace{1cm} (3.2.2)

Finding a reward function $R^*$ becomes equivalent to finding a linear feature weight $\omega^*$ such that

$$\omega^T \mu_E \geq \omega^T \mu(\pi) \hspace{1cm} \forall \pi$$ \hspace{1cm} (3.2.3)

where $\mu_E$ is the empirical accumulated feature value estimated from expert demonstrations. Specifically, given a set of $m$ expert demonstrations $\{s_0^i, s_1^i, \cdots \}_{i=1}^{m}$, we can set

$$\mu_E = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{\infty} \gamma^t \phi(s_t).$$ \hspace{1cm} (3.2.4)

IRL in this setting has the following challenges:

- There can be many different $\omega^*$ that satisfy the inequality (Equation 3.2.3) ($R^* = 0$ from $\omega^* = 0$ is always an optimal reward function). The challenge is known as ambiguity.

- Apart from ambiguity, the solution might be infeasible. There may be no non-trivial $\omega^*$ that satisfies these inequalities (Equation 3.2.3). Expert demonstration might not be sufficient enough to well generalize the optimal policy. The policy $\pi^*$ which gives the demonstration may be sub-optimal that it is within the convex hull of all possible $\pi$’s.
• The problem indeed is very sensitive to imperfect expert demonstrations which may easily result in very biased policy estimation.

• Inequality equations (Equation 3.2.3) need to hold for all possible policies. This makes for very expensive computation.

The original IRL method presented in (Ng et al., 2000) works as following:

1. Initialize $\pi_1$, which is selected randomly.

2. In the $k$-th round, there is a set of $k$ policies \{$\pi_1, \cdots, \pi_k$\}.

3. Obtain $\omega$ via solving the following optimization problem:

$$\max \omega \sum_{i=1}^{k} p(\omega^T \mu_E - \omega^T \mu(\pi_i))$$

s.t. $||\omega|| \leq 1$ (3.2.5) (3.2.6)

where $p(x) = x$ if $x \leq 0$ and $p(x) = 2x$ if $x < 0$. The negative value returned by $p$ function is used to penalize the violation of the assumption that expert demonstrations are optimal:

$$\omega \mu_E \leq \omega \mu(\pi_i), i = 1, \cdots, k.$$ (3.2.7)

4. Solve the reinforcement learning problem with the reward function $R(s) = \omega^T \phi(s)$ which gives a new $\pi_{k+1}$.

5. Repeat step 2 to 4 until a reward function with which we are satisfied.
This algorithm attempts to alleviate the ambiguity issue by choosing \( \omega \) that maximizes the sum of the differences of value functions, and it puts a penalty on the objective function when expert optimality doesn’t hold. There might still be a couple of \( \omega \) that satisfy the maximum of the sum of the differences of value functions. Another issue of this algorithm is that the penalty term is heuristically chosen without reasonable argument.

### 3.2.1 Maximum Margin Planning

Maximum marginal planning (MMP) (Ratliff et al., 2006) has been proposed to solve the ambiguity of choosing a reward function. Maximum marginal planning formulates the IRL problem as:

\[
\begin{align*}
\min_{\omega, \xi} & \quad \| \omega \|^2 + C \sum_i \xi^{(i)} \\
\text{s.t.} & \quad \omega^T \mu^{(i)}_E \geq \omega^T \mu^{(i)}(\pi^{(i)}) + m(\pi^{(i)}_E, \pi^{(i)}) - \xi^{(i)} \quad \forall i, \pi^{(i)}
\end{align*}
\]  

(3.2.7) (3.2.8)

where \( m(\pi^{(i)}_E, \pi) \) term is based on the motivation that a policy that is different from demonstration should have larger margin (e.g. number of different states). The index term \( i \) indicates the constraint formulation of the \( i \)-th MDP setting. A solution via subgradient methods is presented in Ratliff’s paper (Ratliff et al., 2006). We introduce the constraint generation method as follows:

1. Initialize \( \Pi_i \) for all \( i \).
2. Solve

\[
\min_{\omega, \xi} \| \omega \|^2_2 + C \sum \xi^{(i)}
\]

s.t. \( \omega^T \mu_E^{(i)} \geq \omega^T \mu(\pi^{(i)}) + m(\pi_E^{(i)}, \pi^{(i)}) - \xi^{(i)} \) \( \forall i, \forall \pi^{(i)} \in \Pi^{(i)} \).

3. For the current value of \( \omega \), find the most violated constraint for all \( i \) by solving:

\[
\max_{\pi^{(i)}} \omega^T \mu_E^{(i)} + m(\pi_E^{(i)}, \pi^{(i)})
\]

that obtains the optimal policy for the current estimated reward function plus loss augmentation \( m \).

4. For all \( i \) add \( \pi^{(i)} \) to \( \Pi^{(i)} \),

5. Repeat steps 2 to 4 until no constraint violations are found.

MMP solves the ambiguity and infeasibility issues. However, there might be a very large number of active constraints that makes MMP computationally challenging. In addition, the inequality in the constraint (Equation 3.2.8) needs to hold for all expert demonstrations making the prediction performance of MMP very sensitive to imperfect expert demonstrations.
3.2.2 Feature Expectation Matching

Instead of recovering a reward function, Abbeel and Ng (Abbeel and Ng, 2004) proposed a feature-matching approach which attempts to estimate a policy directly with performance close to that of expert’s. The basic motivation is to find a policy $\tilde{\pi}$ such that

$$\mu(\tilde{\pi}) - \mu_E \leq \epsilon.$$  \hfill (3.2.9)

With such inequality, we then for all $\omega$ that $||\omega||_1 \leq 1$, have:

$$||\omega^T \mu(\tilde{\pi}) - \omega^T \mu_E||_2 \leq \epsilon,$$  \hfill (3.2.10)

meaning the feature expectation matching algorithm used to find such a $\tilde{\pi}$ works as following:

1. Random pick a policy $\pi_0$, and set $i = 1$.

2. Find the linear feature weight $\omega$ that expert demonstration maximally outperforms all previously found policies:

$$\max_{\gamma, \omega: ||\omega||^2 \leq 1} \gamma$$

s.t. $\omega^T \mu_E \geq \omega^T \mu(\pi^{(i)}) + \gamma, i = 0, \cdots, i - 1$.

3. If $\gamma \leq \epsilon$, then terminate.
4. Use an optimal control algorithm to compute the optimal policy $\pi^{(i)}$ given reward function

$$R = (\omega^{(i)})^T \phi.$$  

5. Set $i = i + 1$, and go back to step 2.

Finally, $\tilde{\pi}$ is the mixture of some $\pi$‘s returned by the feature expectation matching algorithm. However, the feature values $\mu(\pi)$’s of these policies are in the convex closure of all possible $\mu$‘s, and hence the prediction performance of $\tilde{\pi}$ will have very high variance.

### 3.2.3 Maximum Causal Entropy

Other work (Ziebart et al., 2010) provides an efficient and robust IOC method for fully observable environments via the principle of maximum causal entropy. Building on the Markov-Massey theory of directed information (Massey, 1990), the principle of maximum causal entropy prescribes policy estimation by maximizing entropy over the entire interaction process. Being motivated by maximum entropy, it is straightforward to formulate uncertainty over fully observable interaction process where the dynamic or state transition probability:

$$\{p(s_t|s_{1:t-1}, a_{1:t-1})\}_{t=1}^T$$

is given, and the goal is to estimate the policy:

$$\{p_t(a_t|a_{1:t-1}, s_{1:t})\}_{t=1}^T.$$
Given some expert demonstrations, we can consider to derive the optimal policy from maximizing uncertainty over the entire interaction process by matching expectation constraints:

$$\max_{\{p(s_t|s_{1:t-1}, a_{1:t-1})\}} \mathbb{E}_p \left[-\sum_{t=1}^{T} \log p(a_t|a_{1:t-1}, s_{1:t}) \right]$$

$$\sum_{t=1}^{T} H(a_t|a_{1:t-1}, s_{1:t})$$

Under $\mathbb{E}_p [\mathcal{F}(A_{1:T}, S_{1:T})] = \tilde{\mathbb{E}}_p [\mathcal{F}(S_{1:T}, A_{1:T})].$

Unfortunately, directly solving this optimization problem, a non-convex optimization problem with respect to the policy, is challenging.

This work (Ziebart et al., 2010) treats the entire interaction process to be two interacting temporal processes:

$$p(x_{1:T}, y_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{1:t-1}, y_{1:t-1}) \times \prod_{t=1}^{T} p(y_t|y_{1:t-1}, x_{1:t}).$$

The provided/unknown process are each known as causally conditional probabilities (Kramer, 1998):

$$p(x_{1:T}|y_{1:T-1}) = \prod_{t=1}^{T} p(x_t|x_{1:t-1}, y_{1:t}), \quad p(y_{1:T}|x_{1:T}) = \prod_{t=1}^{T} p(y_t|y_{1:t-1}, x_{1:t}).$$
Definition 3.2. The causal entropy (Kramer, 1998; Permuter et al., 2008) of $Y_{1:T}$ given $X_{1:T}$ is

$$H(Y_{1:T}||X_{1:T}) = \mathbb{E}_p[-\log p(y_{1:T}|x_{1:T})] = \sum_{i=1}^{T} H(Y_i|Y_{1:i-1}, x_{1:i}).$$

Similar to entropy, the causal entropy measures the uncertainty over the entire interaction process with known dynamics and unknown policies.

Though the causal entropy $H(Y_{1:T}||X_{1:T})$ is not a convex function of $\{p(y_t|y_{1:t-1}, x_{1:t})\}_{t=1}^{T}$, it is a convex function of $p(y_{1:T}|x_{1:T})$. This fact motivates the redefinition of the causal conditional probability to a convex optimization problem so that standard convex optimization solution can be adopted to solve the problem.

Definition 3.3. The causal simplex (Ziebart et al., 2013) of $p(y_{1:T}|x_{1:T})$, denoted $\Xi$, is defined by the following causal polytope of affine constraints:

$$\forall x_{1:T} \in \mathcal{X}_{1:T}, y_{1:T} \in \mathcal{Y}_{1:T} \ p(y_{1:T}|x_{1:T}) \geq 0;$$

$$\forall x_{1:T} \in \mathcal{X}_{1:T}, \sum_{y_{1:T} \in \mathcal{Y}_{1:T}} p(y_{1:T}|x_{1:T}) = 1;$$

$$\forall x_{1:T} \in \mathcal{X}_{1:T}, y_{1:T} \in \mathcal{Y}_{1:T}, x'_{1:T} \in \mathcal{X}_{1:T}, y'_{1:T} \in \mathcal{Y}_{1:T}$$

$$\forall \tau \in \{1, \cdots, T\} \ such \ that \ x_{1:\tau} = x'_{1:\tau}$$

$$\sum_{y_{\tau+1:T} \in \mathcal{Y}_{\tau+1:T}} (p(y_{1:T}|x_{1:T}) - p(y_{1:T}|x'_{1:T})) = 0.$$
It shows that the causal conditional probability \( p(y_{1:T}|x_{1:T}) \) defined by the causal simplex is equivalent to being defined by the product of conditional probability.

Similar to entropy, the causal entropy provides a lower bound of the expected number of bits needed to transmit encoded messages from the sequence \( Y_{1:T} \), iteratively over \( t \in \{1, \cdots, T\} \), given the previous \( Y_{1:t-1} \) variables, revealed input \( X_{1:t} \) and excluding unrevealed variables \( X_{t+1:T} \):

\[
\mathbb{E}_p \left[ - \sum_{t=1}^{T} \log p(Y_t|Y_{1:t-1}, X_{1:t}) \right] = \mathbb{E}_p \left[ - \log p(Y_{1:T}|X_{1:T}) \right] \leq \mathbb{E}_p \left[ - \log q(Y_{1:T}|X_{1:T}) \right].
\]

Let

\[
\Gamma = \Xi \cap \{ \mathbb{E}_p [\mathcal{F}(X_{1:T}, Y_{1:T})] = \tilde{\mathbb{E}}_p [\mathcal{F}(X_{1:T}, Y_{1:T})] \}
\]

which is closed and convex, then strong duality of min max optimization holds:

\[
\min_q \max_{p \in \Gamma} \mathbb{E}_q [- \log p] = \max_{p \in \Gamma} \min_q \mathbb{E}_q [- \log p] = \max_{p \in \Gamma} H(Y_{1:T}|X_{1:T}).
\]

Since the causal simplex is a convex set of causal conditional probability, this can then be formulated as a convex optimization problem for policy estimation:

\[
\max_{p \in \Xi} H(Y_{1:T}|X_{1:T})
\]

Under \( \mathbb{E}_p [\mathcal{F}(X_{1:T}, Y_{1:T})] = \tilde{\mathbb{E}}_p [\mathcal{F}(X_{1:T}, Y_{1:T})] \).
The Lagrangian dual form of the convex optimization problem reduces to maximize the causal log likelihood (Ziebart et al., 2013):

$$\max_{\lambda} \mathbb{E} \left[ \log \hat{p}_\lambda(Y_{1:T}||X_{1:T}) \right]$$

$$\hat{p}_\lambda(y_t|x_{1:t}, y_{1:t-1}) = \exp(Q(y_{1:t}, x_{1:t}) - V(y_{1:t-1}, x_{1:t}))$$

$$Q(y_{1:t}, x_{1:t}) = \begin{cases} 
\lambda^T F(x_{1:T}, y_{1:T}) & t = T \\
\mathbb{E}[F(x_{t+1:T}, y_{1:t})|x_{1:t}, y_{1:t}] & t < T 
\end{cases}$$

$$V(y_{1:t-1}, x_{1:t}) = \log \sum_{y_t} \exp(Q(y_{1:t}, x_{1:t})) \quad t \leq T$$

where $\lambda$ is the Lagrangian multiplier. It can be learned via standard gradient descent method (Boyd and Vandenberghe, 2004) with update rule:

$$\lambda_{i+1} = \lambda_i + \eta_i \left( \mathbb{E}[F(X_{1:T}, Y_{1:T})] - \mathbb{E}[F(x_{1:T}, Y_{1:T})] \right).$$

Based on the principle of maximum causal entropy, the optimal policy for the inverse optimal control setting under Markovian state transition dynamics $p(s_t|s_{1:t-1}, a_{1:t-1}) = p(s_t|s_{t-1}, a_{t-1})$ and linear additive feature function $F(s_{1:t}, a_{1:t}) = \sum_{t=1}^T \phi(s_t, a_t)$ has a Markov independence property:

$$\pi_{\theta}(a_t|s_{1:t}, a_{1:t-1}) = \pi_{\theta}(a_t|s_t) = e^{Q(a_t,s_t) - V(s_t)}$$
where

\[ Q(a_t, s_t) = \begin{cases} 
\theta^T \phi(a_T, s_T) & t = T, \\
E[V(s_{t+1}|s_t, a_t)] + \theta^T f(a_t, s_t) & t < T,
\end{cases} \]

and

\[ V(s_t) = \text{softmax}_{a_t} Q(a_t, s_t) = \log \sum_{a_t} e^{Q(a_t, s_t)}. \]
CHAPTER 4

PREDICTIVE IOC FOR LQG

(This chapter includes the paper (Chen and Ziebart, 2015) published as Xiangli Chen and Brian D. Ziebart, “Predictive Inverse Optimal Control for Linear-Quadratic-Gaussian Systems”, in proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, PMLR 38:165-173, 2015.)

Developing a prediction model for process data is an important task for many artificial intelligence applications. It is of key importance for human-robot interaction and human-computer interaction systems. For example, robots that more efficiently and safely navigate around people and user interfaces that can autonomously adapt to improve a user’s task efficiency (Balakrishnan, 2004) each requires accurate process predictions. Perfect accuracy is an unrealistic objective for this task given the large number of process sequences that are possible. Instead, statistical methods are needed to characterize the inherent uncertainties of this prediction task and guide appropriate decision making in artificial intelligence applications.

Direct policy estimation (Pomerleau, 1989) learns a mapping from contextual situations to actions is one of main approaches for constructing predictive models for process prediction task. Unfortunately, as we discuss in section 3.1, policy estimation is not nearly as adaptive to contextual changes in the decision/control setting. Inverse optimal control (IOC) methods (section 3.2) (also known as inverse reinforcement learning and inverse planning) view behavior under a sequential decision process and estimate a reward/cost function that rationalizes
demonstrated behavior (Ramachandran and Amir, 2007; Neu and Szepesvári, 2012; Ziebart et al., 2008; Babes et al., 2011). Often, a learned reward function from the IOC approach generalizes well to other portions of the decision process’s state space and even to other decision processes in which the same reward function is applicable. However, inverse optimal control requires planning, decision, and control problems to be repeatedly solved. This can be computationally expensive. As a result, inverse optimal control methods that have been beneficially employed to estimate decision policies for process prediction tasks have been restricted to settings with low dimensional state-action spaces and/or full state observability (Ziebart et al., 2008; Henry et al., 2010; Ziebart et al., 2012; Levine and Koltun, 2013) to make the optimal control problem tractable.

Unfortunately, the assumptions of a low-dimensional state-control space and full state observability often do not match reality for many important prediction tasks. Often a human actor has only a partial knowledge of the “state of the world” and takes actions that are delayed responses to noisy observations of the actual world state. For example, a person may walk through an environment with occlusions and therefore have uncertainty about the locations of obstacles. Similarly, user interfaces may change in ways that users do not anticipate leading to observed behavior sequences affected by human response times. Extensions of inverse optimal control techniques address either high-dimensionality or partial observability, but not both. IOC methods for high-dimensional data assume a linear-quadratic control setting (Ziebart et al., 2012; Levine and Koltun, 2012), for which optimal control is tractable even for very high dimensional state-control spaces. IOC methods for partially-observable decision
process representations (Choi and Kim, 2011) have been limited to partially-observable Markov decision processes with small state-action spaces.

In this chapter, we introduce our work that extends IOC methods to settings with both high-dimensional state-control spaces and partial observability. We specifically investigate the discrete-time linear-quadratic-Gaussian (LQG) control setting (Section 2.5.2). This is a special sub-class of partially-observable decision processes for which optimal control is efficient even for large state and control dimensions. We formulate the inverse LQG problem from robust estimation first principles to obtain a predictive distribution over state-control sequences. Like the optimal control solution, which combines a Kalman filter (Kalman, 1960) with a linear quadratic regulator (Kwakernaak and Sivan, 1972), our approach finds a similar separation between state estimation and control policy estimation to provide probabilistic predictive distributions. We demonstrate the benefits of incorporating partial observability for predictive inverse optimal control in a synthetic control prediction task and for modeling mouse cursor pointing motions.

4.1 Background and Related Work

4.1.1 Linear-Quadratic-Gaussian Control

Linear-quadratic-Gaussian (LQG) control problems seek the optimal control policy of partially observed linear systems. Section 2.5.2 has given a definition of LQG in the optimal control setting. Here we give a brief introduction of LQG fitting our setting. Apart from its initial value, which is Gaussian distributed (Equation 4.1.1), the unobserved state of the system, denoted as value $x_t$ (or random variable $X_t$) at time $t$, evolves as a noisy linear function of the
previous state $x_{t-1}$ and control $u_{t-1}$ (Equation 4.1.2). The state itself is not directly observed by the controller; instead, observation variables $z_t$ (or as random variables, $Z_t$) that are noisy linear functions of the state are observed (Equation 4.1.3). The state dynamic and observation noise are each conditional Gaussian distributions with a mean defined by linear relationships of the A, B, and C matrices and covariance matrices $\Sigma_{d1}$, $\Sigma_d$ and $\Sigma_o$ characterizing the noise:

\[
X_1 \sim N(\mu, \Sigma_{d1}); \\
X_{t+1}|x_t, u_t \sim N(Ax_t + Bu_t, \Sigma_d); \\
Z_t|x_t \sim N(Cx_t, \Sigma_o).
\]

The independence properties of the LQG control setting are illustrated by the Bayesian network in Figure 4.1.
Given the state and observation dynamics, the LQG optimal control problem is to obtain the control policy \( \pi : \mathcal{U}_{1:t-1} \times \mathcal{Z}_{1:t} \rightarrow \mathcal{U}_t \), that minimizes an expected cost that is quadratically defined in terms of a cost matrix \( M \):

\[
E_f(Z_{t:T+1}, X_{1:T+1}, U_{1:T}) \sum_{t=1}^{T+1} X_t^T M \bar{X}_t .
\]

(4.1.4)

Here we restrict our cost matrix only to the system state \( X_t \). However, extending our prediction method to the general cost matrix (Equation 2.5.8) is straightforward.

Similar to section 2.5, the optimal control law is obtained by separating the problem into a state estimation task and a linear-quadratic-regulation optimal control problem for the estimated state. State estimation is accomplished using a Kalman filter [Kalman, 1960]. Due to the linear characteristics of the problem, only the mean of the state estimate is needed. For the estimate of the state's mean conditioned on previous and current observations and previous controls, \( \hat{x}_t \), the optimal control policy is recursively defined (Stengel, 2012) as

\[
u_t = -L_t \hat{x}_t,
\]

(4.1.5)

\[
\hat{x}_{t+1} = A \hat{x}_t + Bu_t + K_t(z_{t+1} - C(A \hat{x}_t + Bu_t)),
\]

(4.1.6)

\[
\hat{x}_1 = \mathbb{E}[x_1]
\]

(4.1.7)
where $K_t$ is the Kalman gain:

$$K_t = H_t C_T (C H_t C_T + \Sigma_o)^{-1}, \quad (4.1.8)$$

and $H_t$ is determined by the following matrix Riccati difference equation that runs forward in time:

$$H_{t+1} = A(H_t - H_t C_T (C H_t C_T + \Sigma_o)^{-1} C H_t) A_T + \Sigma_d, \quad (4.1.9)$$

$$H_1 = \mathbb{E}[x_1 x_1^T] = \Sigma_{d_1} + \mu \mu^T. \quad (4.1.10)$$

The feedback gain matrix:

$$L_t = (B^T F_{t+1} B)^{-1} B^T F_{t+1} A \quad (4.1.11)$$

where $F_t$ is determined by the following matrix Riccati difference equation that runs backward in time:

$$F_t = \begin{cases} 
M + A^T F_{t+1} A - A^T F_{t+1} B (B^T F_{t+1} B)^{-1} B^T F_{t+1} A & t \leq T, \\
M & t = T + 1.
\end{cases} \quad (4.1.12)$$

The linear-quadratic regulator (LQR) setting (Section 2.5.1) can be viewed as the full-observability special case of LQG with $C = I$ and $\Sigma_o = 0$ where $I$ is the identity matrix so that the observation variable $z_t$ is equivalent to the unobserved state $x_t$. 
4.1.2 Inverse Optimal Control

In contrast with the optimal control problem of obtaining a policy that minimizes some cumulative expected cost, inverse optimal control \cite{Ng2000,Abeel2004} takes (samples from) a policy and tries to obtain a cost function for which observed behavior is optimal, ideally. Early approaches often assumed a linear functional form \cite{Boyd1994,Ratliff2006,Abeel2004} for the cost function in terms of features $f$, $\text{cost}(x_t) = \theta^T f(x_t)$, and optimality for some choice of weights $\theta$. In practice, observed behavior is not consistently optimal for any linear cost function \cite{Abeel2004} for this family of functions, and the optimality assumption breaks down. Even though the linear weights of the function are unknown and behavior can be arbitrarily sub-optimal, any policy that has the same expected feature statistics, $\mathbb{E}[\sum_t f(x_t)]$, as the demonstrated feature statistic expectation is guaranteed to have the same expected cost \cite{Abeel2004}. Mixtures of optimal policies can instead be employed to guarantee the same expected costs as (sub-optimal) demonstrated behavior \cite{Abeel2004}.

We distinguish IOC approaches that match the expected cost of demonstrated behavior with those that attempt to provide predictions for behavior. The principle of maximum entropy has previously been employed for this purpose. Maximum entropy IOC \cite{Ziebart2008} \cite{Section3.2.3} provides a stochastic control policy that robustly minimizes the predictive log-loss for policies that, in expectation, generate certain expected features. In contrast, mixtures of optimal policies \cite{Abeel2004} can produce infinite log-loss when they provide no
support for demonstrated policies. We build upon the maximum entropy IOC approach in this work.

### 4.1.3 Directed Information Theory

We view the LQG setting using concepts and measures from directed information theory (Marko, 1973; Massey, 1990; Kramer, 2003; Tatikonda and Mitter, 2004; Permuter et al., 2008). The joint distribution of states, observations, and controls is factored into two causally conditioned probability distribution ([Kramer, 2003]):

\[
f(x_{1:T}, z_{1:T}, u_{1:T}) = f(u_{1:T} | x_{1:T}, z_{1:T}) \times f(x_{1:T}, z_{1:T} | u_{1:T-1})
\] (4.1.13)

where

\[
f(u_{1:T} | x_{1:T}, z_{1:T}) \triangleq \prod_{t=1}^{T} f(u_t | u_{1:t-1}, x_{1:t}, z_{1:t})
\] (4.1.14)

and

\[
f(x_{1:T}, z_{1:T} | u_{1:T-1}) \triangleq \prod_{t=1}^{T} f(x_t, z_t | x_{1:t-1}, u_{1:t-1}, z_{1:t-1}).
\]

The causal entropy ([Kramer, 2003]) of the LQG control policy:

\[
H(U_{1:T} | X_{1:T}, Z_{1:T}) \triangleq \mathbb{E} [-\log_2 f(U_{1:T} | X_{1:T}, Z_{1:T})] = \mathbb{E} \left[ \sum_{t=1}^{T} H(U_t | X_{1:t}, Z_{1:t}, U_{1:t-1}) \right]
\] (4.1.15)

is a measure of the uncertainty of the causally conditioned probability distribution (Equation [4.1.14]). It can be interpreted as the amount of information present in expectation for a control sequence \(u_{1:T}\) sampled from the joint state, observation, control distribution (Equation [4.1.13]), given only previous observation and control variables.
Due to the specific independence properties of partial observability in the LQG setting (shown in Figure 4.1), the control policy reduces to

\[ f(u_{1:T}|x_{1:T}, z_{1:T}) = f(u_{1:T}|z_{1:T}) \triangleq \prod_{t=1}^{T} f(u_t|u_{1:t-1}, z_{1:t}) \tag{4.1.16} \]

and the causal entropy of the control policy reduces to

\[ H(U_{1:T}|X_{1:T}, Z_{1:T}) = H(U_{1:T}|Z_{1:T}). \]

However, as we shall see, representing the control distribution in its more general form and constraining it to possess the required independence properties is crucial for our approach.

4.2 Inverse Linear-Quadratic-Gaussian Control

We employ a robust estimation approach for learning the control policy in a way that generalizes to different settings (Section 4.2.1). We show that this approach can be posed as a convex optimization problem (Section 4.2.2) leading to a maximum causal entropy problem (Section 4.2.3). The dual solution decomposes into a state estimation component and a (softened) optimal control component enabling efficient inference (Section 4.2.4).

4.2.1 Robust Policy Estimation

We consider a set of policies denoted by \( \Xi \) that are similar to observed sequences of states, observations, and controls (defined precisely in section 4.2.3). We follow the robust estimation formulation (Topsoe, 1979; Grünwald and Dawid, 2004) of maximum entropy inverse optimal control (Ziebart et al., 2010) to select the single policy with the best worst-case predictive
guarantees from this set. This can be viewed as a two-player game in which the policy estimate, 
\( \hat{f} \), that minimizes loss is first chosen, and then an evaluation policy, \( f \in \Xi \), is adversarially 
chosen that maximizes the loss subject to matching known/observed properties of the actual 
policy:

\[
\min_{\{f(u_1:T||z_1:T,x_1:T)\} \ (f(u_1:T||z_1:T,x_1:T))} \max_{\in \Xi} \mathbb{E}_{\hat{f}} [-\log_2 f(U_1:T||X_1:T, Z_1:T)] \quad (4.2.1)
\]

\[
\geq \max_{\{f(u_1:T||z_1:T,x_1:T)\}} \min_{\in \Xi} \mathbb{E}_{\hat{f}} [-\log_2 f(U_1:T||X_1:T, Z_1:T)] \quad (4.2.2)
\]

\[
= \max_{\{f(u_1:T||z_1:T,x_1:T)\} \in \Xi} \mathbb{E}_{f} [-\log_2 f(U_1:T||X_1:T, Z_1:T)]. \quad (4.2.3)
\]

In general, weak Lagrangian duality holds and the dual optimization problem (Equation 4.2.2) 
provides a lower-bound on the primal optimization problem (Equation 4.2.1). The causal log-
loss:

\[
\text{Loss}(\hat{f}, f) = \mathbb{E}_{f} [-\log_2 \hat{f}(U_1:T||X_1:T, Z_1:T)] \quad (4.2.4)
\]

measures the amount of “surprise” (in bits when \( \log_2 \) is used) when control sequences sampled 
from \( f \) are observed while control sequences from \( \hat{f} \) are expected.

When it is employed as the loss function, the dual optimization problem reduces to maxi-
mizing the causal entropy (Equation 4.2.3):

\[
\text{Loss}(f, f) = H(U_1:T||Z_1:T, X_1:T). \quad (4.2.5)
\]
4.2.2 A Convex Definition of the LQG Policy Set

We seek to strengthen our analysis of the dual solution so that primal-dual equality holds (Equation 4.2.2). This strong duality requires the set of policies (Equation 4.1.16) to be convex (Boyd and Vandenberghe, 2004), which is not obviously the case. We introduce the partial observability causal simplex (Definition 4.1), which extends the causal simplex (Ziebart et al., 2010) to the partial-observability setting. It is defined by affine constraints that ensure that members of the set factor according to equation 4.1.16. This is accomplished by preventing unobserved state variables ($x_{1:T}$) and not-yet-revealed observable variables ($z_{t+1:T}$) from influencing a control variable’s distribution ($y_t$).

**Definition 4.1.** The partial observability causal simplex for $f(u_{1:T}||z_{1:T},x_{1:T})$ denoted by $\Delta$, is defined by the following set of constraints:

\[
\forall u_{1:T} \in U_{1:T}, x_{1:T}, x'_{1:T} \in X_{1:T}, z_{1:T}, z'_{1:T} \in Z_{1:T},
\]

\[
f(u_{1:T}||z_{1:T},x_{1:T}) \geq 0, \quad \int_{u'_{1:T} \in U_{1:T}} f(u'_{1:T}||z_{1:T},x_{1:T}) \, du'_{1:T} = 1,
\]

\[
f(u_{1:T}||z_{1:T},x_{1:T}) = f(u_{1:T}||z_{1:T},x'_{1:T}).
\]

\[
\forall \tau \in \{0,\ldots,T\} \text{ such that } z_{1:\tau} = z'_{1:\tau}, x_{1:\tau} = x'_{1:\tau},
\]

\[
\int_{u_{\tau+1:T} \in U_{\tau+1:T}} f(u_{1:T}||z_{1:T},x_{1:T}) \, du_{\tau+1:T} = \int_{u_{\tau+1:T} \in U_{\tau+1:T}} f(u_{1:T}||z'_{1:T},x'_{1:T}) \, du_{\tau+1:T}.
\]

The non-negativity constraints and normalization constraints (Equation 4.2.6) ensure a valid probability distribution. The next set of constraints (Equation 4.2.7) enforces partial observability—the controls do not depend on the hidden state. The final set of constraints
(Equation 4.2.8) ensures that only previous \( x \) and \( z \) variables influence controls \( u \) (causal conditioning). Because all of the equalities and inequalities are affine, the partial observation causal simplex is a convex set.

4.2.3 Maximum Causal Entropy Estimation

Redefining the domain of the estimated policy \( \hat{f}(u_{1:T}||z_{1:T}) \) using the partial observability causal simplex (Definition 4.1) enables strong duality. The dual of the robust policy estimation formulation (Section 4.2.1), reduces to maximizing the causal entropy (Equation 4.1.15) as a selection measure from the set of policies (\( \Delta \)) matching quadratic state expectation constraints (Definition 4.2).

**Definition 4.2.** The maximum causal entropy inverse LQG policy is obtained from:

\[
\arg \max_{\{f(u_{1:T}||z_{1:T},x_{1:T})\} \in \Delta} H(U_{1:T}||Z_{1:T},X_{1:T}) \tag{4.2.9}
\]

such that

\[
\mathbb{E} \left[ \sum_{t=1}^{T+1} X_t X_t^T \right] = \tilde{\mathbb{E}} \left[ \sum_{t=1}^{T+1} X_t X_t^T \right] \tag{4.2.10}
\]

where \( \Delta \) is the partial observability causal simplex of definition 4.1, \( \mathbb{E}[\cdot] \) is the expectation under the estimated policy, and \( \tilde{\mathbb{E}}[\cdot] \) is the empirical expectation from observed behavior sequence data.
This choice of constraints is motivated by inverse optimal control (Section 4.1.2). They ensure that the stochastic control policy matches the performance of observed behavior on unknown state-based quadratic cost functions\footnote{We employ state-based functions for notational simplicity. Control-based functions could also be explicitly added with an additional constraint or implicitly by incorporating an “action memory” into the state vector.} (Corollary 4.1).

**Corollary 4.1.** \cite{AbbeelNg2004}. For any unknown quadratic cost function, parameterized by matrix $M$, matching expected feature counts guarantees equivalent performance on the unknown cost function:

$$\forall M \in \mathbb{R}^{|S| \times |S|}, \quad \mathbb{E} \left[ \sum_{t=1}^{T+1} X_t X_t^T \right] = \mathbb{E} \left[ \sum_{t=1}^{T+1} X_t X_t^T \right] \quad \Rightarrow \quad \mathbb{E} \left[ \sum_{t=1}^{T+1} X_t^T M X_t \right] = \mathbb{E} \left[ \sum_{t=1}^{T+1} X_t^T M X_t \right].$$

Many different mixture distributions over deterministic policies can satisfy this constraint \cite{AbbeelNg2004}. Thus, the causal entropy (Equation 4.2.9) can be viewed as a tie-breaking criterion that resolves the ill-posedness of inverse optimal and provides strong robust prediction guarantees.

### 4.2.4 Predictive Inverse LQG Distribution

The Lagrangian dual form provides a value equivalent solution to the primal constrained optimization problem (Equation 4.2.9), while leading to a more compact representations of the policy.
Theorem 4.1. The solution to the partially-observable maximum causal entropy problem (Definition 4.2) takes the following recursive form where $M$ is the Lagrangian multiplier matrix:

\[
f(u_t|u_{1:t-1}, z_{1:t}) = e^{Q(u_{1:t}, z_{1:t})} - V(u_{1:t-1}, z_{1:t})
\]

where

\[
Q(u_{1:t}, z_{1:t}) = \begin{cases} 
\mathbb{E}[X_{T+1}^{T}MX_{T+1}|u_{1:T}, z_{1:T}] & t = T \\
\mathbb{E}[X_{t+1}^{T}MX_{t+1} + V(U_{t+1}, Z_{t+1})|u_{1:t}, z_{1:t}] & t < T
\end{cases}
\]

\[
V(u_{1:t-1}, z_{1:t}) = \text{softmax} Q(u_{1:t}, z_{1:t}) \triangleq \log \int_{u_t} e^{Q(u_{1:t}, z_{1:t})} du_t.
\]

The probability distribution can be interpreted as a softened relaxation of the Bellman optimal policy criterion (Bellman, 1957) where the softmax function replaces the max function: $\text{softmax}_x f(x) \triangleq \log \int_x e^{f(x)}$. It serves as a smooth interpolator of the maximum function.

Unfortunately, in the LQG setting, the value functions of Theorem 4.1 are still unwieldy since they depend on the entire history of actions and observations. As in optimal LQG control (Kwakernaak and Sivan, 1972), a more practical algorithm is obtained by separating state estimation from the policy distribution. Assuming a Gaussian belief of the current state $X_t|b_t(u_{1:t-1}, z_{1:t}) \sim N(\mu_b, \Sigma_b)$ that is based on the entire history, the policy can be recursively obtained according to Theorem 4.2.
Theorem 4.2. Given a belief state which summarizes the history $u_{t-1:t}, z_{t:t}$ up to time step $t$ (i.e., $X_t|b_t \sim N(\mu_t, \Sigma_t)$), the recurrence values (Equation 4.2.13, 4.2.14) are Markov quadratic functions of the form:

$$Q(u_t, \mu_{b_t}) = \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix}^T W_t \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix}$$

$$V(z_t, u_{t-1}, \mu_{b_{t-1}}) = \begin{bmatrix} z_t \\ u_{t-1} \\ \mu_{b_{t-1}} \end{bmatrix}^T D_t \begin{bmatrix} z_t \\ u_{t-1} \\ \mu_{b_{t-1}} \end{bmatrix}$$

(4.2.15)

$$W_t = \begin{cases} \begin{bmatrix} B & A \end{bmatrix}^T M \begin{bmatrix} B & A \end{bmatrix} + D_{t+1(U\mu,Z)} \begin{bmatrix} CB & CA \end{bmatrix} & t = T \\ \begin{bmatrix} B & A \end{bmatrix}^T M \begin{bmatrix} B & A \end{bmatrix} + D_{t+1(U\mu,Z)} \begin{bmatrix} CB & CA \end{bmatrix} \begin{bmatrix} D_{t+1(Z,Z)} & CB \\ CB & CA \end{bmatrix} & t < T \end{cases}$$

(4.2.16)

$$D_t = P_t^T (W_t(u_t,\mu) - W_{t(U,U)}^{-1} W_t(U,U)) P_t$$

(4.2.17)

where

$$P_t = \begin{bmatrix} E_{t+1} & B - E_{t+1} CB & A - E_{t+1} CA \end{bmatrix}$$

$$E_{t+1} = (\Sigma_d + A\Sigma_{b_t}^T A^T) C^T (\Sigma_o + C(\Sigma_d + A\Sigma_{b_t}^T A^T) C^T)^{-1}.$$
The probabilistic control policy for a belief state with mean \( \mu_{b_t} \) is then:

\[
U_t | \mu_{b_t} \sim N \left( -W_t^{-1} W_t(U, \mu) \mu_{b_t}, -\frac{1}{2} W_t^{-1}(U, U) \right). \tag{4.2.18}
\]

According to the previously developed theory of maximum causal entropy (Ziebart et al., 2010) the gradient of the Lagrangian dual form with respect to the Lagrangian multipliers matrix \( M \) is

\[
E_f \left[ \sum_{t=1}^{T+1} X_t X_t^T \right] - \tilde{E}_f \left[ \sum_{t=1}^{T+1} X_t X_t^T \right]. \tag{4.2.19}
\]

This comes from the constraint of the convex optimization problem (Equation 4.2.10). We can compute the expectation of quadratic state moments over the distribution of state-control-observation trajectories provided by our estimated policy also via conditioning on the mean and variance of the belief state:

\[
E_f \left[ X_t X_t^T \right] = E_{f(u_{1:t-1}, z_{1:t})} \left[ E_{f(x_t | u_{1:t-1}, z_{1:t})} \left[ X_t X_t | U_{1:t-1}, Z_{1:t} \right] \right] \tag{4.2.20}
\]

\[
= E \left[ \Sigma_{b_t} + \mu_{b_t} \mu_{b_t}^T \right] \\
= \Sigma_{b_t} + \text{Var} \left[ \mu_{b_t} \right] + E \left[ \mu_{b_t} \right] E \left[ \mu_{b_t} \right]^T.
\]

The mean and variance of the belief state \( \mu_{b_t}, \Sigma_{b_t} \) is recursively computed according to a Kalman filter (Kalman, 1960). So the standard gradient descent method can be adopted to learn Lagrangian multipliers matrix \( M \).
Theorem 4.3 establishes the connection to optimal control: the mean/mode of the control distribution is the optimal control and, in fact, the (stochastic) maximum causal entropy probabilistic control policy can be directly obtained from the optimal control solution.

**Theorem 4.3.** The terms of the stochastic control policy (Equation 4.2.18) are related to the LQG optimal control laws as

\[
W_t(U,U) = B^T F_{t+1} B, \quad W_t(U,\mu) = B^T F_{t+1}
\]  

(4.2.21)

where \( F_{t+1} \) is defined by the optimal control law (Equation 4.1.12), and the Lagrangian multiplier matrix \( M \) in equation 4.2.16 is given as the cost matrix in equation 4.1.12.

Thus, existing methods for solving LQG optimal control problems can be used to recover the stochastic policy given the cost matrix \( M \).

### 4.3 Experiments

We evaluate the performance of our inverse LQG approach on synthetic data (Section 4.3.1) and real mouse cursor movement data (Section 4.3.2) to investigate its benefits in comparison to full observability models of behavior by average empirical causal log loss over test data:

\[
-\frac{1}{N} \sum_{n=1}^{N} \log f(u_{1:T_n} || z_{1:T_n}) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log f(u_t | u_{1:t-1}, z_{1:t}) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \log f(u_t | \mu_{b_t})
\]

The empirical causal log loss is also named as trajectory log loss.
4.3.1 Synthetic Experiment

The synthetic experiment investigates the benefits of incorporating partial observability into predictive inverse optimal control. This provides some insights into whether it is sufficient to simply ignore partial observability and use inverse optimal control models that assume full observability.

The data is collected via an optimal LQG controller applied to a spring mass system:

\[
X_{t+1} = AX_t + BU_t + \varepsilon_s \quad Z_t = CX_t + \varepsilon_o
\]

\[
X_1 \sim N(0, \Sigma_{d1}) \quad \varepsilon_x \sim N(0, \Sigma_d) \quad \varepsilon_o \sim N(0, \Sigma_o).
\]

\[
A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}.
\]

The controller minimizes the following expected quadratic cost function:

\[
J = \mathbb{E} \left[ \sum_{t=1}^{T+1} X_t^T Q X_t \right]
\]

where (using \( I \) as the identity matrix):

\[
Q = I_{2 \times 2} \quad \Sigma_{d1} = \Sigma_d = \sigma_d \ast I_{2 \times 2} \quad \Sigma_o = \sigma_o \ast I_{1 \times 1}.
\]

From the setting of the observation dynamic \( C \), only the first row of the two row state \( X_t \) is observed which provides partial-observability scenario. Each experiment where the noise of
the system varies generates 2000 state observation control trajectories with length \( T = 30 \) by applying the optimal LQG controller. The first 1000 trajectories are training data, and the remaining 1000 trajectories are the testing data. Having \( C = I_{2 \times 2} \) and \( Z_t = X_t \) simulates the LQR model via LQG setting. Figure 4.2 left, shows that LQG has significantly better performance than LQR when the observation noise \( \sigma \) increases. This is because the controller is basing its controls on noisy observations that are increasingly different from the true state. Also, it shows that the log loss decreases as the observation noise increases. This is because as the observation noise increases, the controllers system state estimates are less certain. Paying large costs for controls becomes less worthwhile, and controls from a smaller range (closer to 0) are instead produced. These lower variance controls are easier to predict and have small log loss. As shown in Figure 4.2 right, when the state noise \( \sigma_d \) increases, it dominates the test performance of both LQG and LQR. This is because as the state noise increases, state estimation becomes increasingly error prone for both LQG and LQR.

4.3.2 Mouse Cursor Experiment

The mouse cursor data is captured from 20 non-motor-impaired computer users performing computer cursor pointing tasks to assess the benefits of the LQG approach versus previous LQR models that have been employed for this task (Ziebart et al., 2012). Users are presented with a sequence of circular clicking targets to select and their mouse cursor data is collected at 100Hz. The experiment investigates whether incorporating a response delay using LQG framework provides better predictions than the LQR approach, which assumes an instantaneous response to the changes in mouse cursor position. The assumption is instead that due to the imprecise
human abilities for fine-grained control, cursor navigation is essentially an open loop control problem and that incorporating feedback delay will produce better policy estimates. Some evidence of this is demonstrated by the cursor pointing trajectories in Figure 4.3.

The control formulation follows the previous work \cite{Ziebart2012}. The instantaneous state:

\[
x_t \triangleq [x_t \ y_t \ \dot{x}_t \ \dot{y}_t \ \ddot{x}_t \ \ddot{y}_t]^{\top}
\]
Figure 4.3. Example mouse cursor trajectories terminating at small circle positions exhibiting characteristics of delayed feedback.

is represented by the relative position, velocity, and acceleration vectors towards the target and orthogonal to the target at discrete points in time. These dynamics (e.g., velocities and accelerations) are defined according to difference equations:

\[
\begin{pmatrix}
\dot{x}_t \\
\dot{y}_t \\
\ddot{x}_t \\
\ddot{y}_t
\end{pmatrix}
= \begin{pmatrix}
x_t - x_{t-1} \\
y_t - y_{t-1} \\
\dot{x}_{t-1} - \dot{x}_{t-2} \\
\dot{y}_{t-1} - \dot{y}_{t-2}
\end{pmatrix} \quad \begin{pmatrix}
\ddot{x}_t \\
\ddot{y}_t
\end{pmatrix}
= \begin{pmatrix}
\dot{x}_t - \dot{x}_{t-1} \\
\dot{y}_t - \dot{y}_{t-1}
\end{pmatrix}
\] (4.3.1)

and can easily be expressed as a linear dynamics model with the control vector \(u_t\) representing the change in position. Under this dynamics model, mouse pointing motion data follows a linear relationship (with optional zero-mean Gaussian noise, \(\epsilon\)):

\[x_{t+1} = Ax_t + Bu_t + \epsilon\]
where

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

The experiment considers a control system with a delayed observation of \( t_0 \) time step. This is formally represented in the LQG model by augmenting the LQG state with the previous \( t_0 \) states and having the observation dynamics only reveal the state from \( t_0 \) time steps ago. For example, a delay one model has the following dynamics matrices:

$$A' = \begin{bmatrix} A & 0 \\ I & 0 \end{bmatrix} \quad \quad B' = \begin{bmatrix} B \\ 0 \end{bmatrix} \quad \quad C' = \begin{bmatrix} 0 & I \end{bmatrix}.$$

We additionally compare our prediction approach against a direct policy estimation method (Section 3.1): \( k^{th} \)-order Markov models of different orders \( k = \{1, 2, 3, 4\} \). For this continuous state-action setting, estimating the Markov model reduces to a linear regression problem of the form:

$$\hat{s}_t = [s_{t-1} \ s_{t-2} \ \ldots \ s_{t-k}]\alpha + \epsilon \quad (4.3.2)$$
with zero-mean Gaussian noise $\epsilon \sim N(0, \sigma^2)$. The state at each time is the $x$ and $y$ position of the mouse cursor. Regression parameters $\alpha$ are estimated by minimizing the sum of squared errors, as is standard in ordinary linear regression. Control estimates $\hat{u}_t$ are simply the difference between the next state estimate, $\hat{s}_t$ and the previous state, $s_{t-1}$, with the distribution determined by the Gaussian model.

Figure 4.4. Average trajectory log-loss of: the LQG model with various amounts of delay, $t_0$; the LQR model; Markov models of order 2,3,4.

Of the 4,949 mouse cursor trajectories, we randomly select 3,000 as training data and use the remaining 1,949 for evaluation. In Figure 4.4 we evaluate different choices of delayed
feedback, $t_0$. We have no observation noise. Note that the model is equivalent to the LQR setting when $t_0 = 0$. As shown in this figure, the LQG setting with $t_0 = 3$ delay has the best performance. The Markov models of $3^{rd}$ and $4^{th}$ order outperform the LQR model, but are not more predictive than the LQG model with delay of 1, 2, 3, or 4. (Note that $1^{st}$ order Markov model is significantly worse and does not appear in the figure.) The advantage of the LQG model over the noise-less LQR inverse optimal control model and the direct policy estimation of the Markov model shows that modeling mouse pointing motions as an LQG problem is advantageous compared to the previous LQR model, which assumes instantaneous responses.

4.4 Summary

LQG control is a work horse for solving optimal control problems characterized by partial observability. In this chapter, we extended maximum entropy inverse optimal control to the LQG control setting. We established a separation property that allows inference in the resulting model to be performed efficiently. Despite the formulation of our approach being distinct from optimal LQG control, we found close connections between the two methods, including the ability to use an LQG solver as an integral part of the inverse LQG inference procedure. We demonstrated the advantages of the LQG representation for predictive inverse optimal control both on synthetic datasets and on a real mouse cursor dataset.
CHAPTER 5

ROBUST COVARIATE SHIFT REGRESSION

(This chapter includes the paper (Chen et al., 2016a) published as Xiangli Chen, Mathew Monfort, Anqi Liu and Brian D. Ziebart, “Robust Covariate Shift Regression”, in proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics, PMLR 51:1270-1279, 2016.)

Linear regression (Section 3.1.1) is prevalently employed across the data sciences to understand relationships between variables and to make predictions (Friedman et al., 2001; Bishop, 2006; Murphy, 2012). It is commonly assumed, in various ways, that the labeled source data available to train the linear regression model (pairs of input vectors and output scalars) is representative of the target data that it will use for making predictions (given only input vectors). However, in many datasets, this assumption is not valid; source data can often come from biased portions of the input space and measured relationships often violate the linearity assumption for the output variable made by the linear regression model. One form of this problem, where source data is biased and potentially non-representative, is known as covariate shift (Section 2.6). It occurs when the (stochastic) mapping from inputs to output is shared by the source and target data (i.e., source/target data is distributed according to $f_{src}(x)f(y|x)$ and $f_{trg}(x)f(y|x)$, but the distribution of inputs can vary ($f_{src}(x) \neq f_{trg}(x)$). Existing methods for addressing covariate shift employ importance weighting (Section 2.6.1) in an attempt to reweight available source data so that it may serve as an unbiased estimate for functions.
of the target data (Shimodaira, 2000). Unfortunately, when the amount of covariate shift is substantial and the number of source samples is small, importance weighting often leads to very high variance estimates (Cortes and Mohri, 2014) and inaccurate regression models. We illustrate the fundamental problems faced in regression under covariate shift using the Hahn1 dataset shown in Figure 5.1 as a running example in this chapter. Locally, the datapoints appear linear in many portions of this dataset. A biased sample of datapoints from strictly within those regions will often mislead existing importance weighting methods into incorrectly linearly extrapolating beyond the data.

Figure 5.1. Hahn1 taste representing the result of a National Institute of Standards and Technology (NIST) study of the thermal expansion of copper.
Drawing on a recently developed method for robustly learning classifiers from data that has sample selection bias (Liu and Ziebart, 2014), in this chapter, we propose a novel robust approach for regression in the covariate shift setting as a significant first step to deal with covariate shift for general process prediction tasks. Our approach assumes that the stochastic mapping from inputs to the output variable is as similar as possible to a “zero-knowledge” reference distribution, except where statistics measured from source data indicate otherwise. We develop our robust bias-aware regression approach and illustrate its behavior on this dataset before evaluating it using additional higher-dimensional datasets.

5.1 Related Work

Domain adaptation tasks of learning from a source domain and predicting in a target domain have been an area of significant investigation (Daumé and Marcu, 2006; Ben-David et al., 2007; Blitzer et al., 2008; Glorot et al., 2011; Zhang et al., 2013; Cortes and Mohri, 2014), but it is an inherently difficult task (Ben-David et al., 2010). We focus on the special case of covariate shift (Shimodaira, 2000; Zadrozny, 2004; Dudik et al., 2005; Fan et al., 2005; Huang et al., 2006; Sugiyama et al., 2008; Blitzer et al., 2011; Yu and Szepesvári, 2012; Sugiyama and Kawanabe, 2012; Wen et al., 2014; Reddi et al., 2015), in which the conditional distribution of the predicted variable is shared in both source and target domains. In this section, we review the least squares linear regression formulation, existing covariate shift methods for regression, and robust minmax methods leveraged by our approach.
5.1.1 Least Squares Linear Regression

Ordinary least squares (OLS) regression assumes a linear relationship between input variables \( x \) and output variable \( y \), \( \hat{y}_{a,b}(x) = b^T x + a \), that is parameterized by weights \( a \) (scalar) and \( b \) (vector). A standard method for estimating these parameters is to minimize the sum of squared residuals between estimator \( \hat{y}_{a,b}(x_i) \) and the actual output variable \( y_i \) for each example \( i \). This is equivalent to an expected squared loss over the empirical distribution \( \tilde{f}(x) \tilde{f}(y|x) \):

\[
\arg\min_{a,b} \mathbb{E}_{f(x)f(y|x)} \left[ (Y - \hat{y}_{a,b}(X))^2 \right]. \tag{5.1.1}
\]

This optimization (Equation 5.1.1) is equivalent to maximizing the conditional log likelihood of sample data:

\[
\arg\max_{a,b,\sigma} \mathbb{E}_{f(x)f(y|x)} \left[ \log \hat{f}_{a,b,\sigma}(Y|X) \right] \tag{5.1.2}
\]

with \( \hat{f}(y|x) \) normally distributed with mean \( \hat{y}_{a,b}(x) \) and variance \( \sigma^2 \) [Bishop, 2006]. This corresponds to a zero-mean Gaussian residual error model with variance \( \sigma^2 \).

5.1.2 Importance Weighted Linear Regression

Under covariate shift, the input of source data for estimating the linear regression model comes from a distribution \( f_{src}(x) \) that differs from the target data input distribution \( f_{trg}(x) \) on which it will be employed to make predictions. OLS under covariate shift (Equation 5.1.1) does not minimize the residual error (equivalently maximize the likelihood) of data drawn from the target distribution, \( f_{trg}(x)f(y|x) \), and it is not a consistent estimator. Importance weighted
least squares (IWLS) (Sugiyama and Kawanabe, 2012) is often employed to reweight the source data by the importance ratio, $f_{\text{trg}}(x)/f_{\text{src}}(x)$, to estimate the target distribution’s residual error, which is then minimized:

$$\arg\min_{a,b} \mathbb{E}_{f_{\text{src}}(x)f(y|x)} \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)}(Y - \hat{y}_{a,b}(X))^2 \right].$$ \hspace{1cm} (5.1.3)

As we discuss in section 2.6.1, this provides consistent estimates that minimize the target residual error asymptotically:

$$\lim_{n \to \infty} \min_{a,b} \mathbb{E}_{f_{\text{src}}^{(n)}(x)f(y|x)} \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)}(Y - \hat{y}_{a,b}(X))^2 \right] = \min_{a,b} \mathbb{E}_{f_{\text{trg}}(x)f(y|x)} [(Y - \hat{y}_{a,b}(X))].$$ \hspace{1cm} (5.1.4)

However, finite sample generalization bounds require the importance ratio’s first moment to be finite (Cortes et al., 2010), $\mathbb{E}_{f_{\text{src}}(x) \left[ f_{\text{trg}}(X)/f_{\text{src}}(X) \right]} < \infty$. Unfortunately, many biased source distributions violate this requirement (e.g., when the target distribution has larger variance than the source distribution). Estimates that result from reweighting under those distributions typically have high variance, as a small number of source datapoints with large importance weights, $f_{\text{trg}}(x)/f_{\text{src}}(x)$, determine the regression model’s parameters.
5.1.3 Adaptive Importance Weighted Linear Regression

In practice, the adaptive importance weighted least squares (AIWLS) method, which is a slightly stabilized variant of IWLS, is often preferable (Sugiyama and Kawanabe, 2012). Following equation [5.1.2], the model maximizes the weighted likelihood:

\[
\operatorname{argmax}_{a,b,\sigma} \mathbb{E}_{f_{\text{src}}(x)} f_{\text{src}}(y|x) \left[ \left( \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \right)^{\gamma} \log \hat{f}_{a,b,\sigma}(Y|X) \right]
\] (5.1.5)

where \( \gamma \in (0,1) \) is the flattening parameter. It balances the estimator’s consistency and stability, with ordinary least squares (\( \gamma = 0 \)), and importance weighted least squares (\( \gamma = 1 \)) at its extremes.

5.1.4 Robust Minimax Learning

Minimax robust estimation (Topsoe, 1979; Grünwald and Dawid, 2004) prescribes the predictor that minimizes the worst-case prediction loss (Asif et al., 2015; Wang et al., 2015). When the logarithmic loss (log loss), \( \mathbb{E}_{f(x)f(y|x)} \left[ -\log \hat{f}(y|x) \right] \), is employed as the loss function, the minimax robust estimation approach reduces to the principle of maximum entropy (Jaynes, 1982):

\[
\max_{f(y|x)} H(Y|X) \triangleq \mathbb{E}_{f(x)f(y|x)} [-\log f(Y|X)]
\] (5.1.6)

such that: \( \mathbb{E}_{f(x)f(y|x)} [\Phi(X,Y)] = \mathbb{E}_{f(x)f(y|x)} [\Phi(X,Y)] \).
This principle is a foundational method for deriving many familiar exponential family distributions (e.g., Gaussian, exponential, Laplacian, logistic regression, conditional random fields (Lafferty et al., 2001)) by incorporating constraints of various known statistics, \( \Phi(\cdot) \) (Wainwright and Jordan, 2008). The OLS model, \( \hat{y} = a + b^T x \), results from robustly minimizing the log-loss in the independent and identically distributed (IID) setting, subject to matching quadratic interaction features: \( \Phi(x, y) = \text{vector}([y x^T 1]^T[y x^T 1]) \).

A recently developed approach for learning under sample selection bias investigates learning probabilistic classifiers when the entropy measure of equation \[5.1.6\] is evaluated according to the target distribution, \( f_{\text{trg}}(x) \), and the constraints of the maximum entropy optimization are expectations over the source distribution (Liu and Ziebart, 2014). We build on this approach by extending it to predict continuous-valued variables using a relative entropy (Kullback-Leibler divergence) formulation that avoids distribution degeneracies.

### 5.2 Robust Bias-Aware Regression

#### 5.2.1 Minimax Estimation Formulation

A natural loss function to consider is the conditional log-loss on the target distribution:

\[
\text{log-loss}_{f_{\text{trg}}(x)}(f(y|x), \hat{f}(y|x)) \triangleq \mathbb{E}_{f_{\text{trg}}(x)f(y|x)}[-\log \hat{f}(Y|X)]. \tag{5.2.1}
\]

This conditional log-loss measures the amount of expected “surprise” (in bits) to see samples from \( f_{\text{trg}}(x)f(y|x) \) when samples are assumed to come from \( f_{\text{trg}}(x)\hat{f}(y|x) \) instead (Cover and Thomas, 2006). As described in section \[5.1.4\] when the source and target distribution match (or
their differences are ignored), the minimax robust estimation approach with this loss function subject to quadratic interaction feature constraints yields the ordinary least squares solution to regression. Thus, using this loss function as a starting point can be viewed as a natural generalization of standard linear regression methods.

We define the difference in conditional log-loss between an estimator \( \hat{f}(y|x) \) and a baseline conditional distribution \( f_0(y|x) \) on the target data distribution \( f_{\text{trg}}(x)f(y|x) \) as the relative loss:

\[
\text{rel-loss}_{f_{\text{trg}}(x)}(f(y|x), \hat{f}(y|x), f_0(y|x)) \\
= \log \text{-loss}_{f_{\text{trg}}(x)}(f(y|x), \hat{f}(y|x)) - \log \text{-loss}_{f_{\text{trg}}(x)}(f(y|x), f_0(y|x)) \\
= E_{f_{\text{trg}}(x)f(y|x)} \left[ -\log \frac{\hat{f}(Y|X)}{f_0(Y|X)} \right]. \tag{5.2.2}
\]

It measures the amount of expected relative “surprise” of data from \( f_{\text{trg}}(x)f(y|x) \) assumed to come from \( f_{\text{trg}}(x)\hat{f}(y|x) \) instead of \( f_{\text{trg}}(x)f_0(y|x) \).

We consider the setting in which the conditional distribution \( f(y|x) \) is known to satisfy certain statistical properties (denoted by set \( \Xi \)):

\[ \Xi \triangleq \{ f(y|x) \mid E_{f_{\text{src}}(x)f(y|x)}[\Phi(X,Y)] = c \} \tag{5.2.3} \]

where \( c = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i, y_i) \) is a vector of statistics measured from training data. We seek the regression model estimator \( \hat{f}(y|x) \) that is most robust to the “most surprising” distribution that can arise from covariate shift (formally expressed by definition 5.1).
Definition 5.1. The **robust bias-aware regression estimator**, \( \hat{f}(y|x) \), is the saddle point solution of the following minimax optimization:

\[
\min_{f(y|x)} \max_{f(y|x) \in \Xi} \text{rel-loss}_{f_{\text{reg}}}(f(y|x), \hat{f}(y|x), f_0(y|x)).
\]

This minimax optimization can be interpreted as a two-player game in which the estimator first chooses \( \hat{f}(y|x) \) to minimize relative loss, and then the adversarial evaluation player chooses \( f(y|x) \) to maximize relative loss. Note that both conditional probabilities are also constrained to the conditional probability simplex, denoted \( \Delta \):

\[
\forall x \in \mathcal{X}, y \in \mathcal{Y}, f(y|x) \geq 0; \quad \forall x \in \mathcal{X}, \int_y f(y|x) \, dy = 1.
\]

Theorem 5.1. The solution of the minimax relative log-loss optimization (Definition 5.1) minimizes the target distribution conditional Kullback-Leibler divergence:

\[
D_{f_{\text{reg}}(x), f(y|x)} (\hat{f}(Y|X)||f_0(Y|X)) \triangleq \mathbb{E}_{f_{\text{reg}}(X)f(Y|X)} \left[ \log \frac{\hat{f}(Y|X)}{f_0(Y|X)} \right]
\] (5.2.4)

subject to matching statistics, \( c \), on the source distribution:

\[
\min_{f(y|x) \in \Delta} D_{f_{\text{reg}}(x), f(y|x)} (\hat{f}(Y|X)||f_0(Y|X))
\] (5.2.5)

such that: \( \mathbb{E}_{f_{\text{src}}(x)f(y|x)} [\Phi(X,Y)] = c \).
Figure 5.2. The conditional mean (solid blue line) and 95% confidence interval (CI, dashed dot magenta line) of least squares linear regression (LS), importance weighted least squares (IWLS) and robust bias-aware regression (RBA) via KL-divergence learned via 90 biased source samples (red cross) and evaluated on 118 target datapoints (black point) of the Hahn1 dataset.

We note that the objective function of this optimization is convex and the constraints are each affine. Thus, standard tools from convex optimization can be employed to obtain the solution to the constrained optimization. We establish the parametric solution to the optimization problem in Theorem 5.2.

**Theorem 5.2.** The robust bias-aware regression for target distribution $f_{\text{trg}}(x)$ estimated using constraints from the source distribution $f_{\text{src}}(x)$ takes the form:

$$
\hat{f}_{\theta}(y|x) \propto f_{o}(y|x)e^{-\frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)}}\theta^{T}\Phi(x,y)
$$

(5.2.6)
with parameters obtained via target distribution maximum conditional log likelihood estimation:

\[
\theta = \arg \max_{\theta} \mathbb{E}_{f_{\text{src}}(x)f(y|x)} \left[ \log \hat{f}_{\theta}(Y|X) \right].
\] (5.2.7)

The distribution’s certainty is moderated by the density ratio, \( f_{\text{src}}(x)/f_{\text{trg}}(x) \). Note that it is the inverse of the importance weight, \( f_{\text{trg}}(x)/f_{\text{src}}(x) \) from equation 5.1.3. As the density ratio \( f_{\text{src}}(x)/f_{\text{trg}}(x) \) goes to zero, the estimator \( \hat{f}(y|x) \), converges to the baseline conditional probability \( f_0(y|x) \). As the ratio goes towards infinity, the estimator converges to a deterministic point estimate. The behavior of the robust approach is shown in Figure 5.2. Where source data is sparse, its uncertainty increases substantially to provide a more uncertain—and in this case, better—fit to the non-linearity of the underlying dataset. In contrast, least squares (LS) and importance-weighted least squares (IWLS) estimate their extrapolative uncertainty based only on the (reweighted) source data, yielding over-confident predictions for the target distribution.

5.2.2 Parameter Estimation

Optimizing equation 5.2.7 appears to be difficult because samples from \( f_{\text{trg}}(x)f(y|x) \) are unavailable to estimate the target distribution log likelihood. However, the form of \( f_\theta(y|x) \) prescribed by the robust bias-aware approach (Theorem 5.2), enables the gradient of the target likelihood function to be computed efficiently using the source distribution (Theorem 5.3).
Theorem 5.3. The gradient of the conditional log likelihood estimation takes the following form:

$$\nabla \theta \mathbb{E}_{f_{	ext{src}}(x)f(y|x)} \left[ \log \hat{f}_\theta(Y|X) \right] = \mathbb{E}_{f_{	ext{src}}(x)f_\theta(y|x)} [\Phi(X,Y)] - c.$$ (5.2.8)

Unfortunately, computing the needed expectations can be difficult when arbitrary feature functions $\Phi(\cdot, \cdot)$, are employed. By restricting consideration to quadratic feature functions and conjugate baseline conditional distributions $f_0(y|x)$, a distribution $\hat{f}(y|x)$ with tractable normalization and expectation computations is obtained (Corollary 5.1). As noted in section 5.1.4, this set of quadratic features is also the basis for standard least squares linear regression under a robust logarithmic loss formulation. Thus, the representational power in terms of features is equivalent to OLS and IWLS variants (Section 5.1.2).

Corollary 5.1. If the base distribution is conditional Gaussian, $f_0(y|x) = N(\mu_o, \sigma_o^2)$, and the feature function has a quadratic form, $\Phi(x,y) = [y \ x^T] \ 1^T [y \ x^T \ 1]$ (where $\theta$ is the vectorized matrix $M$):

$$\theta^T \text{vector}(\Phi(x,y)) = M \cdot \begin{bmatrix} y \\ x \\ 1 \end{bmatrix}^T \begin{bmatrix} y \\ x \\ 1 \end{bmatrix} = \begin{bmatrix} y \\ x \\ 1 \end{bmatrix}^T M \begin{bmatrix} y \\ x \\ 1 \end{bmatrix}$$
then the robust bias-aware regression is also a conditional Gaussian distribution:

\[
\hat{f}_M(y|x) \sim N(\mu(x, M), \sigma^2(x, M))
\]  

(5.2.9)

where:

\[
M = \begin{bmatrix}
M_{(y,y)} & M_{(y,x_1)} \\
M_{(x_1,y)} & M_{(1,1)}
\end{bmatrix}
\]

\[
\mu(x, M) = \left( \frac{2f_{\text{src}}(x)}{f_{\text{trg}}(x)} M_{(y,y)} + \frac{1}{\sigma_o^2} \right)^{-1} \begin{bmatrix} x \\ 1 \end{bmatrix} + \frac{1}{\sigma_o^2} \mu_o
\]  

(5.2.10)

\[
\sigma^2(x, M) = \left( \frac{2f_{\text{src}}(x)}{f_{\text{trg}}(x)} M_{(y,y)} + \frac{1}{\sigma_o^2} \right)^{-1}
\]  

(5.2.11)

Correspondingly, by Theorem 5.2, model parameters are selected by maximizing the target log likelihood:

\[
M = \arg\max_M \mathbb{E}_{f_{\text{trg}}(x) f(y|x)} \left[ \log \hat{f}_M(Y|X) \right]
\]  

(5.2.12)

with gradient:

\[
\mathbb{E}_{f_{\text{src}}(x) f_M(y|x)} [\Phi(X, Y)] - c.
\]  

(5.2.13)

The base distribution plays an important role in our robust regression approach. A simple and straightforward way to set up the base distribution is to assume that it is a Gaussian
distribution $N(\mu_o, \sigma_o^2)$ with mean and variance estimated from the range $[y_{\text{min}}, y_{\text{max}}]$ of $y$’s of the source dataset $D_{\text{src}}$:

$$
\mu_o = \frac{y_{\text{min}} + y_{\text{max}}}{2}, \quad \sigma_o^2 = \left(\frac{y_{\text{max}} - \mu_o}{2}\right)^2.
$$

(5.2.14)

Hence all of the $y$’s of the source dataset are located within the 95% confidence of the base distribution.

Due to the convexity [Boyd and Vandenberghe, 2004] of the robust formulation (Theorem 5.1), convergence to a global optimum is guaranteed using standard gradient-based methods.

We solve the optimization problem with the quadratic feature function defined in Corollary 5.1 by its dual form (Equation 5.2.12). Computing the gradient (Equation 5.2.13) requires taking the expectation over the source distribution, which is challenging. Instead, we use the empirical expectation over the source dataset, which approximates the real expectation in the constraints:

$$
\mathbb{E}_{f_{\text{src}}(x)} f_{M(y|x)} [\Phi(X,Y)] \approx \mathbb{E}_{f_{\text{src}}}(y|x_i) \cdot \Phi(x_i, Y).
$$

Additionally, only estimates of source distribution statistics,

$$
\tilde{c} \triangleq \mathbb{E}_{f_{\text{src}}(x)} \tilde{f}(y|x) \cdot \Phi(X,Y) = \frac{1}{N} \sum_{i=1}^{N} \Phi(x_i, y_i),
$$

(5.2.15)

from sample data are used to constrain the estimated conditional probability distribution, $\tilde{f}(y|x)$, rather than exact source distribution statistics, $c = \mathbb{E}_{f_{\text{src}}(x)f(y|x)} [\Phi(X,Y)]$. This intro-
duces finite sample error that should be accounted for in the parameter estimation. One way to accomplish this is by relaxing the source distribution constraints to incorporate some slack, $\epsilon$: $||E_{f_{\text{src}}(x)f(y|x)}[\Phi(X,Y)] - \tilde{c}|| \leq \epsilon$. Primal relaxed constraints correspond with regularization of the conditional log likelihood maximization in the dual (Altun and Smola, 2006). Following Theorem 5.3, the gradient under $\ell_2$-regularization is:

$$E_{f_{\text{src}}(x)f(y|x)}[\Phi(X,Y)] - \tilde{c} - \lambda M = \frac{1}{N} \sum_{i=1}^{N} E_{f_{M}(y|x_i)} [\Phi(x_i, Y)] - \tilde{c} - \lambda M$$

where $\lambda$ is the regularization weight.

Algorithm 5.1 shows the batch gradient descent method for robust bias-aware regression.

---

**Algorithm 5.1** Batch gradient descent method for robust bias-aware regression

**Input:** source dataset $\{(x_i, y_i)\}_{i=1}^{N}$, source/target distributions $f_{\text{src}}(x)$, $f_{\text{trg}}(x)$, feature function $\Phi(x, y)$, statistics $\tilde{c}$, learning rate $\gamma$, convergence threshold $\tau$ and regularization weight $\lambda$.

**Output:** Model parameters $M$

initialize $M$

repeat

for each source data example $i$ do

$\mu(x_i, M) \leftarrow \left( 2 \frac{f_{\text{src}}(x_i)}{f_{\text{trg}}(x_i)} M_{(y, 1)} + \frac{1}{\sigma^2} \right)^{-1}$

$\sigma^2(x_i, M) \leftarrow \left( 2 \frac{f_{\text{src}}(x_i)}{f_{\text{trg}}(x_i)} M_{(y, y)} + \frac{1}{\sigma^2} \right)^{-1}$

end for

$\nabla L \leftarrow \frac{1}{N} \sum_{i=1}^{N} E_{f_{M}(y|x_i)} [\Phi(x_i, Y)] - \tilde{c} - \lambda M$

$M \leftarrow M - \gamma \nabla L$

until $||\nabla L|| \leq \tau$

return $M$
5.2.3 Relation with Other Methods

Robust-bias aware regression (RBA) is a general framework that reduces to maximizing
the relative target conditional entropy under certain constraints. Due to a lack of target data,
constraints are usually chosen to match feature expectations under the source distribution.
However, it could be helpful to manipulate the constraints and incorporate feature match-
ing under a different distribution based on some side information, corresponding to different
assumptions. We assume the statistics provided by expert knowledge or computed under a
generalized distribution are \(c'\). When we incorporate the strong assumption that the feature
expectation under the target distribution is equivalent to the expectation of reweighted features
on source data, RBA is equivalent to IWLS (Theorem 5.4).

**Theorem 5.4.** When a target-distribution-based constraint:

\[
E_{f_{\text{trg}}(x)f(y|x)}[\Phi(X,Y)] = c' \triangleq E_{\tilde{f}_{\text{src}}(x)f(y|x)} \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \Phi(X,Y) \right]
\]

is applied, and the feature function takes the quadratic form of Corollary 5.1, RBA regression
is equivalent to IWLS regression.

This indicates that IWLS is a special case under the general robust bias-aware regression
framework.

Figure 5.2(c) may appear similar to a Bayesian treatment of the regression problem (e.g.,
a Gaussian process). We establish a key difference between our approach and the comparable
Bayesian technique, Bayesian linear regression model (BLR), here. By letting $\theta^T = [b^Ta]$ and $\hat{y}_\theta(x) = [x^T\theta]$, the linear regression model becomes:

$$f(y|x) = \hat{y}_\theta(x) + \epsilon \text{ where } \epsilon \sim N(0,\sigma^2). \quad (5.2.16)$$

If we treat $\sigma^2$ as a known constant, BLR (Bishop, 2006) assumes a prior distribution for $\theta$. The multivariate Gaussian conjugate prior, $\theta \sim N(\mu,\Sigma)$, is commonly applied. Given source data $(x_i,y_i)_{i=1}^N$, the posterior distribution of $\theta$ is inferred by Bayes’ rule:

$$f(\theta|X,y) = N(A^{-1}(\Sigma^{-1}\mu + \sigma^{-2}X^Ty),A^{-1})$$

where $X$ is a $N \times (d+1)$ design matrix with each row $[x_i^T,1]$, $y = [y_1,y_2,...,y_n]^T$ and $A = \sigma^{-2}X^TX + \Sigma^{-1}$. The predictive distribution $f(y|x_t,X,y)$ for target datapoint $x_t$ is given by averaging the output of all possible linear models with respect to the posterior, yielding a conditional Gaussian distribution $N(\mu(x_t),\Sigma(x_t))$:

$$\mu(x_t) = [x_t^T1]A^{-1}(\Sigma^{-1}\mu + \sigma^{-2}X^Ty) \quad (5.2.17)$$

$$\Sigma(x_t) = [x_t^T1]A^{-1}[x_t^T1]^T + \sigma^2. \quad (5.2.18)$$

Thus, as the amount of source data increases, BLR converges to the OLS regression model (Bishop, 2006), minimizing source distribution squared loss rather than target distribution.
squared loss. In contrast, RBA provides nonlinear prediction means that robustly minimize target log-loss with uncertainty in its distribution that is moderated by the density ratio.

A straightforward extension of RBA for continuous-valued variables is to build RBA via conditional differential entropy $H(Y|X)$. Following the similar procedure built for RBA via KL-divergence in section 5.2, the resulting conditional Gaussian distribution of RBA$_{DE}$ has the following form:

$$
\mu(x, M) = -(M_{(y,y)})^{-1} \begin{pmatrix} M_{(y,x)} \\ x \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix}
$$

$$\sigma^2(x, M) = \left( 2 \frac{f_{src}(x)}{f_{trg}(x)} M_{(y,y)} \right)^{-1}.
$$

We note that when the density ratio $f_{src}(x)/f_{trg}(x)$ goes to zero, the predictor will give very large log-loss due to predictions produced that have very large uncertainty (variance). In contrast, our proposed RBA via KL-divergence (RBA$_{KLD}$) converges to the base distribution $f_{o}(y|x)$. A base distribution provides an upper bound of log-loss of RBA$_{KLD}$. 

5.3 Experiments

5.3.1 Datasets

We employ publicly available regression datasets from the UCI repository (Bache and Lichman, 2013) to evaluate our approach. The number of examples and features, and a basic description of the output of each dataset are listed in Table 1.
### TABLE I

Datasets for empirical evaluation

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Examples</th>
<th>#Features</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airfoil</td>
<td>1503</td>
<td>5</td>
<td>sound pressure</td>
</tr>
<tr>
<td>Concrete</td>
<td>1030</td>
<td>8</td>
<td>strength</td>
</tr>
<tr>
<td>Housing</td>
<td>506</td>
<td>14</td>
<td>value of home</td>
</tr>
<tr>
<td>Music</td>
<td>1059</td>
<td>66</td>
<td>latitude</td>
</tr>
<tr>
<td>Crime</td>
<td>1994</td>
<td>127</td>
<td>crime rate</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>5725</td>
<td>16</td>
<td>UPDRS score</td>
</tr>
<tr>
<td>WineQuality</td>
<td>6497</td>
<td>11</td>
<td>quality score</td>
</tr>
<tr>
<td>IndoorLocation</td>
<td>21048</td>
<td>529</td>
<td>latitude</td>
</tr>
</tbody>
</table>

5.3.2 Constructing Dataset with Bias

We consider experimental settings with both synthetically created bias and naturally occurring bias. The amounts of source and target data, and the type of bias for each experiment are listed in Table II.

We first evaluate our approach on datasets with artificially created bias between the source and target distributions. This allows us to show the generalizability of the proposed method on a larger number of controlled experiments. Given a dataset $D = \{x_i, y_i\}_{i=1}^{n}$, we create biased source $D_{src}$ and target $D_{trg}$ datasets using the following sampling procedure:

1. Split $D$ randomly and evenly into two disjoint datasets $D_1$ and $D_2$.
2. Compute the sample mean $\bar{x}$ and sample covariance $Q$ of $D_1$. 

TABLE II

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Source</th>
<th>#Target</th>
<th>Bias Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airfoil</td>
<td>150-751</td>
<td>752</td>
<td>synthetic</td>
</tr>
<tr>
<td>Concrete</td>
<td>100-515</td>
<td>515</td>
<td>synthetic</td>
</tr>
<tr>
<td>Housing</td>
<td>75-253</td>
<td>253</td>
<td>synthetic</td>
</tr>
<tr>
<td>Music</td>
<td>160-529</td>
<td>530</td>
<td>synthetic</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>1430-2862</td>
<td>2863</td>
<td>synthetic</td>
</tr>
<tr>
<td>Crime</td>
<td>40-278</td>
<td>1716-1954</td>
<td>different state</td>
</tr>
<tr>
<td>WineQuality</td>
<td>4898</td>
<td>1599</td>
<td>different color</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>1877</td>
<td>1839</td>
<td>different age</td>
</tr>
<tr>
<td>IndoorLocation</td>
<td>9371</td>
<td>10566</td>
<td>different floor</td>
</tr>
</tbody>
</table>

3. Construct a Gaussian distribution \(N(\bar{x}, Q)\) and sample a datapoint \(x_{seed}\) to construct another Gaussian distribution \(N(x_{seed}, \alpha Q)\) with bias weight \(\alpha \in [0, 1]\). We employ \(\alpha = 0.3\) in our experiments.

4. Sample from \(D_1\) in proportion to \(N(x_{seed}, \alpha Q)\) to get the biased source dataset \(D_{src}\).

5. Let \(D_2\) be the target dataset \(D_{trg}\).

Our sampling procedure is motivated by the purpose of learning from a biased source dataset and comparing prediction performance on an unbiased target dataset.

To highlight the benefits of our method in practice, we also conduct experiments under naturally occurring bias. The crime dataset is separated into pairs of source and target data where the source only contains data for a single state and the target contains data for all other states combined. We average the results of training with each state as the source and the
other states as the target. The Parkinsons dataset is separated by different age ranges. The source data includes all samples of subjects whose age is below 59 while the target dataset holds samples with subjects ranging in age from 60 to 69. For the wine quality dataset, we train our model on white wine samples as the source dataset, and then use red wine samples as the target dataset. Finally, we evaluate our approach on a large, high-dimensional dataset: the IndoorLocation dataset. We use data collected from low floors as source data and data collected from high floors as target data.

5.3.3 Density Estimation

Multivariate Gaussian Kernel density estimation (Bishop, 2006) is a popular density estimation method that converges to the true probability density of samples asymptotically. Unfortunately it suffers from the curse of dimensionality and is not reliable in high-dimensional problems. Importance weighted methods have been proposed where logistic regression is used to directly form the estimations (Sugiyama and Kawanabe, 2012). By applying Bayes theorem, the importance ratio can be written as:

$$\frac{f_{\text{trg}}(x)}{f_{\text{src}}(x)} = \frac{n_{\text{src}}}{n_{\text{trg}}} \exp \left( \omega^T \begin{bmatrix} x \\ 1 \end{bmatrix} \right)$$

where $\omega$ is selected to maximize the (regularized) likelihood of source and target datapoints, which need not be labeled. We refer the reader to previous work (Qin, 1998; Cheng and Chu, 2004; Bickel et al., 2007; Sugiyama and Kawanabe, 2012) for a full description of the derivation for this approach.
Figure 5.3. Five plots of the average empirical logloss of seven methods for target datasets with 95% confidence interval with amounts of source data. A bar figure showing empirical log loss on four natural bias datasets.

5.3.4 **Comparison Approaches**

We compare our approach, robust-bias aware regression via KL-divergence (RBA_{KLD}), with six other regression methods, which each produce (conditional) Gaussian estimates. We evaluate the empirical log-loss of each model on target data $D_{trg}$: $\frac{1}{n_{trg}} \sum_{i=1}^{n_{trg}} \log f(y_i | \mu(x_i), \sigma^2(x_i))$. The methods are: The baseline (BS) Gaussian distribution $Y \sim N(\mu_o, \sigma_o^2)$ is independent of input $x$ with mean and variance estimated from source data (Equation 5.2.14).
Robust bias-aware regression using differential entropy (RBA_{DE}) yields the conditional Gaussian distribution $Y|X \sim N(\mu(x, M), \sigma^2(x, M))$ with mean and variance defined by equation 5.2.19 and equation 5.2.20 respectively, and $M$ estimated via, e.g., gradient descent.

Bayesian linear regression (BLR) provides the conditional Gaussian distribution $Y|X \sim N(\mu(x), \Sigma(x))$ with mean and variance specified by equation 5.2.17 and equation 5.2.18 respectively. In practice, it is challenging to get the prior information for a Bayesian approach. Here we assume the prior distribution of $\theta$ follows a Gaussian distribution $N(0, I)$ where $I$ is the identity matrix. The variance $\sigma^2$ of noise $\epsilon$ (Equation 5.2.16) is the same as the optimal estimator of variance of LS.

LS, IWLS and BAIWLS share the common formulation of the conditional Gaussian distribution $Y|X \sim N(b^T x + a, \sigma^2)$. The linear parameters $(b, a)$ of LS are estimated by equation 5.1.2 that is to maximize the log-likelihood of source data which also gives the optimal estimator of $\sigma^2$ (Bishop, 2006). Being different from LS, the linear parameters $(b, a)$ and the variance $\sigma^2$ of adaptive importance weighted least squares (AIWLS) are estimated by maximizing adaptive reweighted log-likelihood from equation 5.1.5 which is equivalent to minimizing adaptive reweighted residual error (Equation 5.1.5) when estimating $(b, a)$. The argument is very similar to the case of LS. The linear parameters $(b, a)$ and the variance $\sigma^2$ of IWLS are given by equation 5.1.5 with $\gamma = 1$. For BAIWLS, we consider the minimum empirical log-loss over target dataset $D_{trg}$ achieved by choosing the optimal flattening parameter $\gamma$ from $\Upsilon = \{0.1, 0.2, \ldots, 0.8, 0.9\}$. 
5.3.5 Results

As shown in Figure 5.3, RBA\textsubscript{KLD} has the smallest average empirical log loss, and it provides better performance than the other methods on almost all experiment settings. The result varies as the amount of synthetic bias source data increases, which highlights the motivation of RBA\textsubscript{KLD}. When there is only a small amount of source data that is significantly biased from the target data, the performance of LS, IWLS and BAIWLS are much worse than RBA\textsubscript{KLD} of which the performance guided by the baseline is still sound. When more of the source data is given, and the covariate shift still exists, RBA\textsubscript{KLD} makes significant improvement from the baseline and still performs better than nearly all other methods even if they have better performance than the baseline in some cases. When the amount of source data gets close to the target so that the degree of covariate shift is small, the performance of RBA\textsubscript{KLD} converges to the performance of the other linear regression methods.

As for BAIWLS, we apply a whole range of flattening parameter $\gamma$ (0.1, 0.2, ..., 0.8, 0.9) in each experiment and choose the one which gives the minimum empirical log-loss each time. Even though this unrealistically generous choice of the flattening parameter is given to BAIWLS, it turns out that the flattening parameter does not improve much with respect to LS and IWLS.

BLR has better performance than LS, BAIWLS and IWLS. The reason may be that the prior assumption in BLR compensates for the biased results from learning the source. But, as discussed in section 5.2.3, BLR converges to LS as the amount of source data increases. Shown in the bar figure of Figure 5.3, the performance of BLR in the experimental settings with natural bias and large number of source data is as bad as LS.
RBA_{DE} has competitive performance in some of the experimental settings. But, it has the worst performance in the experimental setting of Parkinsons with synthetic bias, and it has even worse performance than BS in the experimental settings of Crime and Parkinsons with natural bias. As discussed in section 5.2.3, the reason may be that RBA_{DE} will lead to very large log loss in wide support if the target data is very biased from the source one that the density ratio in equation 5.2.20 goes to zero.

5.4 Summary

We proposed a novel minimax approach for regression problems under covariate shift. The minimax approach minimizes relative loss in the worst case, and reduces to minimizing conditional Kullback-Leibler divergence. Due to continuous-valued prediction variables in the regression case, we restrict our constraint to quadratic feature functions and use conditional Gaussian baseline distributions, leading to a conditional Gaussian regression model. We compared the proposed robust method with a range of existing regression models on both synthetically created and natural bias experimental settings of a range of real regression datasets including large size and high dimensional datasets to show its benefit under covariate shift.
CHAPTER 6

ADVERSARIAL IMITATION LEARNING FRAMEWORK

(This chapter includes the paper (Chen et al., 2016b) published as Xiangli Chen, Mathew Monfort, Brian D. Ziebart and Peter Carr, “Adversarial Inverse Optimal Control for General Imitation Learning Losses and Embodiment Transfer”, in proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence, 2016.)

Inverse optimal control (IOC) (Kalman, 1964; Rust, 1988; Boyd et al., 1994a) and inverse reinforcement learning (IRL) (Ng et al., 2000; Abbeel and Ng, 2004) (Section 3.2) attempt to rationalize demonstrated sequential decision making by estimating a reward/cost function that makes observed decision sequences optimal. When the learned reward is defined over abstract properties of states and actions (Ng et al., 2000), it can generalize to new decision processes with states and actions that are similarly described. In contrast, methods that directly estimate a policy mapping from states to controls—also known as “behavioral cloning” (Pomerleau, 1989) (Section 3.1)—often generalize poorly when attempting to predict goal-directed sequential decisions when aspects of the decision process change.

Unfortunately, the basic IOC problem—selecting a reward function that makes demonstrated decision sequences optimal—is ill-posed, since degenerate solutions exist (e.g., setting all rewards to zero makes every decision sequence optimal) (Ng et al., 2000) (Section 3.2). When demonstrated behavior is noisy, only degenerate solutions may remain as valid solutions to the basic IOC problem. Existing methods pose the problem in various ways to avoid degenerate
solutions. Maximum margin planning (MMP) (Ratliff et al., 2006) (Section 3.2.1) seeks a reward function that makes demonstrated sequences have larger reward than all alternatives by a structured loss measure. Maximum (causal) entropy IRL (Ziebart et al., 2010) (Section 3.2.3), and its extensions (Boularias et al., 2011; Levine et al., 2011), seek an entropy-maximizing distribution over sequences/policies that matches the feature-based components of the reward function with demonstrated sequences. Each method is constructed around a specific loss function: MMP minimizes the structured hinge loss, while MaxEnt IRL minimizes the (causal) log loss.

A typical assumption in IOC is that the demonstrator and the learner operate under identical decision processes. In other words, it is assumed that the demonstrator and imitator utilize the same action space, and have the same state transition dynamics. In such settings, imitation can be effectively accomplished by adequately predicting what a demonstrator would do in a new situation. We consider generalized imitation learning problems where the abilities of the demonstrator and the learner are different. This situation arises frequently in practice due to differences in embodiment between human demonstrators and robotic imitators (Nehaniv and Dautenhahn, 2002; Alissandrakis et al., 2002), and, more generally, when autonomously-controlled devices are more expensive and less capable than manually-controlled devices.

In this chapter, we propose a more general framework for inverse optimal control that is both consistent and computationally practical for a range of loss functions and situations where imitation learning across different embodiments is required. The key philosophy of our approach is that unknown properties of how the demonstrator would
behave in new situations should be treated as pessimistically as possible, since any unwarranted assumptions could lead to substantial errors when behavior is evaluated under more general loss functions or transferred across embodiments. Our formulation produces a zero-sum game between: the learner seeking a control policy to minimize loss; and an adversary seeking a control policy that adequately characterizes the demonstrations, but maximizes the learner’s loss. We establish consistency and generalization guarantees, develop algorithms for inference and learning under this formulation, and illustrate the benefits of this approach on synthetic and real imitation learning tasks.

6.1 Background and Notation

Decision processes are defined by state and action sets \( (S, A) \) and the state transition dynamics \( \tau \), which describe the distribution of next states \( s_{t+1} \in S \) given current state \( s_t \in S \) and action \( a_t \in A \): \( \tau(s_{t+1}|s_t, a_t) \). We make use of causally conditioned probability distributions \cite{Kramer1998}, \( P(y_{1:T}|x_{1:T}) \triangleq \prod_{t=1}^{T} P(y_t|y_{1:t-1}, x_{1:t}) \) to compactly represent a decision process’s state transition dynamics

\[
\tau(s_{1:T}|a_{1:T-1}) \triangleq \prod_{t=1}^{T} \tau(s_t|s_{1:t-1}, a_{1:t-1}),
\]

and stochastic control policy

\[
\pi(a_{1:T}|s_{1:T}) \triangleq \prod_{t=1}^{T} \pi(a_t|a_{1:t-1}, s_{1:t}).
\]
Multiplied together, these produce a joint probability distribution over the states and actions:

\[ P(a_{1:T}, s_{1:T}) = \pi(a_{1:T} \mid| s_{1:T})\tau(s_{1:T} \mid| a_{1:T-1}). \]

We denote deterministic control policies (a special case of stochastic control policies) mapping from states or state histories to actions as \( \delta(s_t) \) or \( \delta(s_{1:t}) \). In addition to denoting the demonstrator’s full control policy, \( \pi \), under dynamics, \( \tau \), we also consider distributions of trajectories sampled from the demonstrator’s distribution as \( \tilde{\pi}, \tilde{\tau} \), and a learner’s control policy, \( \hat{\pi} \), under a (possibly different) set of dynamics \( \hat{\tau} \), and estimates of the demonstrator’s policy, \( \hat{\pi} \). We similarly denote states, actions, and deterministic policies corresponds to these different sources as \( \tilde{s}, \hat{s}, \tilde{s}, \hat{a}, \hat{\delta} \), etc.

### 6.2 Problem Definition

We begin by formally defining the supervised learning task of imitation learning with general loss measures in definition 6.1. The learner’s performance is measured by a loss function relating the expected state sequence of the learned control policy with the state sequence resulting from the demonstrator’s control policy. The key inductive reasoning challenge is for the learner to produce a good control policy when demonstrations are unavailable by appropriately inferring the demonstrator’s behaviors in such situations.

**Definition 6.1.** In the task of *imitation learning with general losses and embodiments*, at training time: demonstrated traces of behavior are available from distribution \( \hat{P}(A_{1:T}, S_{1:T}) \) under a dynamical system with known dynamics, \( \tau(S_{1:T} \mid| A_{1:T}) \), and unknown control policy...
Figure 6.1. Learning to imitate a slower robot capable of walking over barriers.

\[ \pi(A_{1:T}||S_{1:T}) \]. The learner attempts to choose a control policy \( \hat{\pi}(\hat{A}_{1:T}||\hat{S}_{1:T}) \) for potentially different dynamics, \( \hat{\tau}(\hat{S}_{1:T}||\hat{A}_{1:T}) \), that, at testing time, minimizes a given loss function relating (unknown) demonstration policies and learned policies: \( \min_{\hat{\pi}} \text{loss}_{\hat{\tau}, \hat{\pi}}(\pi, \hat{\pi}) \).

When the demonstrator and the learner operate under different state-action transition dynamics, \( \tau \neq \hat{\tau} \), we refer to this setting as the imitation learning across embodiments problem. We assume that a loss function expressing the undesirability of the imitator’s differences from the demonstrator is available. The key challenge is that the learner must still estimate the control policy of the demonstrator to be able to generalize to new situations, while also constructing its own control policy to overcome its differences in embodiment. We show a simple illustrative example of this in Figure 6.1.
The ability to minimize the desired imitative loss function when provided enough demonstration data and a sufficiently expressive characterization of decision policies is desired in an imitation learning algorithm. This is formally known as Fisher consistency (Definition 6.2).

**Definition 6.2.** An imitation learning algorithm producing policy $\pi_{imit}$ is **Fisher consistent** if, given the demonstrator’s control policy for any demonstrator/imitator decision processes, $(\tau, \hat{\tau})$, and a sufficiently expressive feature representation for policies, the policy $\pi_{imit}$ is a loss minimizer:

$$\pi_{imit} \in \arg\min_{\hat{\pi}} \mathbb{E}\left[\text{loss}_{\tau, \hat{\tau}}(\pi, \hat{\pi})\right]. \quad (6.2.1)$$

We focus our attention in this work on loss functions that additively decompose over the state sequence\(^1\)

$$\mathbb{E}_{P(a_{1:T}, s_{1:T}, \hat{a}_{1:T}, \hat{s}_{1:T})}\left[\sum_{t=1}^{T} \text{loss}(S_t, \hat{S}_t) \right| \pi, \tau, \hat{\pi}, \hat{\tau}]$$

where the state-action distribution is obtained by combining a stochastic control policy with a state-transition dynamics distribution:

$$P(a_{1:T}, s_{1:T}, \hat{a}_{1:T}, \hat{s}_{1:T}) = \tau(s_{1:T}||a_{1:T-1})\pi(a_{1:T}||s_{1:T})\hat{\tau}(\hat{s}_{1:T}||\hat{a}_{1:T-1})\hat{\pi}(\hat{a}_{1:T}||\hat{s}_{1:T}). \quad (6.2.2)$$

\(^1\)Loss functions for state-action pairs can also be incorporated by defining new states that (partially) “remember” previous state-action histories.
Important to this problem definition is the independence between the demonstrator and the learner: there is no direct influence of one’s actions on the other’s state or actions, as shown in the factorization of the joint distribution.

6.3 Adversarial Approach

We develop an adversarial approach to the problem of imitation learning with general losses and embodiments (Definition 6.1) by combining the idea of rationalizing demonstrated behaviors from inverse optimal control (Abbeel and Ng, 2004) with a game-theoretic perspective (Topsøe, 1979; Grünewald and Dawid, 2004) that incorporates different imitative losses. Our approach assumes that except for certain properties of the limited samples of available demonstrated behavior, the demonstrator’s policy is the worst-case possible for the learner. This avoids generalizing from available demonstrations in an optimistic manner that may be unrealistic and ultimately detrimental to the learner. Using tools from convex optimization (Theorem 6.3) and constraint generation (Algorithm 6.1), this formulation can be solved efficiently (Algorithm 6.2). Though the demonstrator’s true policy is unlikely to be maximally detrimental to the learner, considering it as such leads to Fisher consistency (Theorem 6.1), provides strong generalization guarantees (Theorem 6.2), and avoids making any unwarranted assumptions.

6.3.1 Adversarial Formulation and Properties

Our approach employs a game-theoretic formulation of the prediction task for additive state-based losses. We introduce an adversarially-estimated policy, \( \hat{\pi} \), which must be similar to demonstrated training data traces, but is the worst-case for the learner otherwise, as formalized in definition 6.3.
Definition 6.3. The adversarial inverse optimal control learner for the joint demonstrator/learner transition dynamics, \((\tau, \hat{\tau})\) is defined as a zero-sum game in which each player chooses a stochastic control policy, \(\hat{\pi}\) or \(\check{\pi}\), optimizing:

\[
\min_{\hat{\pi}} \max_{\check{\pi} \in \tilde{\Xi}} \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \check{S}_t) \left| \hat{\pi}, \tau, \check{\pi}, \hat{\tau} \right] \right. \tag{6.3.1}
\]

where \(\tilde{\Xi}\) represents a convex set of constraints measured from characteristics of the demonstrated data (e.g., the moment-matching constraints):

\[
\hat{\pi} \in \tilde{\Xi} \iff \mathbb{E} \left[ \sum_{t=1}^{T} \phi(\hat{S}_t) \left| \hat{\pi}, \tau \right] = \tilde{c} \triangleq \mathbb{E} \left[ \sum_{t=1}^{T} \phi(S_t) \left| \check{\pi}, \hat{\tau} \right] \right. \tag{6.3.2}
\]

of inverse reinforcement learning \((\text{Abbeel and Ng, 2004})\) and the joint state-action distributions are realized by combining control policy and state-transition dynamics: e.g.,

\[
P(\hat{a}_{1:T}, \hat{s}_{1:T}) = \hat{\pi}(\hat{a}_{1:T}||\hat{s}_{1:T})\check{\tau}(\hat{s}_{1:T}||\hat{a}_{1:T-1}). \tag{6.3.3}
\]

Though maximum margin methods, such as MMP \((\text{Ratliff et al., 2006})\) in the imitation learning setting, can similarly incorporate arbitrary structured loss functions, they are not Fisher consistent (Definition 6.2) even for the relatively simple Hamming loss (i.e., number of
state mismatches between two sequences). We establish the consistency of the adversarial inverse optimal control approach in Theorem 6.1.

**Theorem 6.1.** Given a sufficiently rich feature representation defining the constraint set $\Xi$, the adversarial inverse optimal control learner is a Fisher consistent loss function minimizer for all additive, state-based losses.

**Proof.** A sufficiently rich feature representation is equivalent to the constraint set $\Xi$ containing only the true policy $\pi$. Then, under $\tilde{\pi} = \pi$, equation (6.3.1) reduces to:

$$\min_{\hat{\pi}} \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \tilde{S}_t) \bigg| \pi, \tau, \hat{\pi}, \tilde{\tau} \right] \tag{6.3.4}$$

which is the loss function minimizer. \qed

An additional desirable property of this approach—even when the feature representation is not as expressive—is that if the set $\tilde{\Xi}$ can be defined to include the demonstrator’s true policy, $\pi$, then generalization performance will be upper bounded by the expected adversarial training loss (Theorem 6.2).

**Theorem 6.2.** The adversarial transfer IOC formulation (Definition 6.3) provides a useful generalization bound: if the true demonstrator policy $\pi$ resides within the constraint set $\tilde{\Xi}$ with

---

1This follows directly from the Fisher inconsistency of multiclass classification [Liu, 2007; Tewari and Bartlett, 2007] using the Crammer-Singer multi-class generalization of the hinge loss [Crammer and Singer, 2002], which is a special case of the imitation learning setting with a single time step.
probability at least $1 - \alpha$, then the generalization error will be worse than the training error (expected game value) with probability no more than $\alpha$:

$$P(\pi \in \tilde{\Xi}) \geq 1 - \alpha \implies P\left( E \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, S_t)|\pi, \tau, \hat{\pi}, \hat{\tau} \right] \geq E \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t)|\hat{\pi}, \tau, \hat{\pi}, \hat{\tau} \right] \right) \leq \alpha. \quad (6.3.5)$$

Proof. If $\pi \in \tilde{\Xi}$, then:

$$E \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, S_t)|\pi, \tau, \hat{\pi}, \hat{\tau} \right] \leq E \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t)|\hat{\pi}, \tau, \hat{\pi}, \hat{\tau} \right],$$

since replacing $\pi$ with a worst case policy (maximizer of the set), $\hat{\pi}$, only makes the expected loss worse. Thus, bounds on $P(\pi \in \tilde{\Xi})$ provide generalization guarantees with at least as much probability.

6.3.2 Learning and Inference Algorithms

Building on recently developed methods for tractably solving adversarial prediction problems for classification with cost-sensitive (Asif et al., 2015) and multivariate (Wang et al., 2015) performance measures, we employ the method of Lagrange multipliers to simplify from a game with one player’s actions jointly constrained to a parameterized game with only probabilistic constraints on each player’s policy (Theorem [6.3]).
Theorem 6.3. An equilibrium for the game of definition 6.3 is obtained by solving an unconstrained zero-sum game parameterized by a vector of Lagrange multipliers:

\[
\min_w \min_{\hat{\pi}} \max_{\tilde{\pi}} \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \tilde{S}_t) + w \cdot \phi(\tilde{S}_t) \right| \hat{\pi}, \tau, \hat{\pi}, \tilde{\pi} ] - w \cdot \tilde{c}.
\]

Proof. The proof follows from applying the method of Lagrangian multipliers (a) to the constrained optimization problem of equation 6.3.1 and then employing strong Lagrangian duality and minimax duality (b):

\[
\min_{\hat{\pi}} \max_{\tilde{\pi}} \min_w \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \tilde{S}_t) \right| \hat{\pi}, \tau, \hat{\pi}, \tilde{\pi} ] \tag{a}
= \min_w \min_{\hat{\pi}} \max_{\tilde{\pi}} \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \tilde{S}_t)w \cdot \left( \sum_{t=1}^{T} \phi(\hat{S}_t) - \tilde{c} \right) \right| \hat{\pi}, \tau, \hat{\pi}, \tilde{\pi} ] \tag{b}
= \min_{\hat{\pi}} \min_{\tilde{\pi}} \max_w \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \tilde{S}_t) + w \cdot \phi(\tilde{S}_t) \right| \hat{\pi}, \tau, \hat{\pi}, \tilde{\pi} ] - w \cdot \tilde{c}.
\]

Note that we assume that the loss function is an expected loss over state predictions. The objective function of our optimization is therefore a bilinear function of the learner’s strategy and the adversary’s strategy, which provides the strong Lagrangian duality that we employ. No stronger assumption about the state-based loss function is needed so long as it takes this bilinear form. \qed
We form the stochastic policy of each player \( \pi, \hat{\pi} \) as a mixture of deterministic policies: \( \delta \) and \( \hat{\delta} \). Conceptually, the payoff matrix of the zero-sum game can be constructed by specifying each combination of deterministic policies, \( \delta, \hat{\delta} \), having payoff:

\[
\mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t) + w \cdot \phi(\hat{S}_t)|\delta, \tau, \hat{\delta}, \hat{\tau} \right].
\]  

(6.3.6)

An example payoff matrix is shown in Table III with the adversary choosing a distribution over columns, and the learner choosing a distribution over rows.

<table>
<thead>
<tr>
<th></th>
<th>( \delta_1 )</th>
<th>( \delta_2 )</th>
<th>\ldots</th>
<th>( \delta_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \delta_1 )</td>
<td>( \ell(\delta_1, \delta_1) )</td>
<td>( \ell(\delta_2, \delta_1) )</td>
<td>\ldots</td>
<td>( \ell(\delta_k, \delta_1) )</td>
</tr>
<tr>
<td></td>
<td>+( \psi(\delta_1) )</td>
<td>+( \psi(\delta_2) )</td>
<td></td>
<td>+( \psi(\delta_k) )</td>
</tr>
<tr>
<td>( \delta_2 )</td>
<td>( \ell(\delta_1, \delta_2) )</td>
<td>( \ell(\delta_2, \delta_2) )</td>
<td>\ldots</td>
<td>( \ell(\delta_k, \delta_2) )</td>
</tr>
<tr>
<td></td>
<td>+( \psi(\delta_1) )</td>
<td>+( \psi(\delta_2) )</td>
<td></td>
<td>+( \psi(\delta_k) )</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
<td>\vdots</td>
<td>\ldots</td>
<td>\vdots</td>
</tr>
<tr>
<td>( \hat{\delta}_j )</td>
<td>( \ell(\delta_1, \hat{\delta}_j) )</td>
<td>( \ell(\delta_2, \hat{\delta}_j) )</td>
<td>\ldots</td>
<td>( \ell(\delta_k, \hat{\delta}_j) )</td>
</tr>
<tr>
<td></td>
<td>+( \psi(\delta_1) )</td>
<td>+( \psi(\delta_2) )</td>
<td></td>
<td>+( \psi(\delta_k) )</td>
</tr>
</tbody>
</table>

TABLE III

The payoff matrix for the adversarial IOC prediction game with \( \ell(\delta, \hat{\delta}) = \mathbb{E}[\sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t)|\delta, \tau, \hat{\delta}, \hat{\tau}] \) and \( \psi(\hat{\delta}) = \omega \cdot \mathbb{E}[\sum_{t=1}^{T} \phi(\hat{S}_t)|\delta, \tau] \).

Unfortunately, this leads to a payoff matrix with a size that is exponential in terms of the actions in the decision process. This cannot be explicitly constructed for practical problems of
even modest size. We employ the double oracle method (McMahan et al., 2003) to construct a smaller sub-portion of the matrix that supports the Nash equilibrium strategy for the full game. The basic strategy, outlined in Algorithm 6.1, iteratively computes a Nash equilibrium for a payoff sub-matrix and augments the payoff matrix with an additional column and row that provide the most improvement for each player.

Algorithm 6.1 Double oracle method for adversarial IOC

Input: Demonstrator’s state transition dynamics $\tau_D$; learner’s state transition dynamics, $\hat{\tau}$; loss function: $\text{loss}(s_t, \hat{s}_t)$; initial policy sets: $\Pi$ and $\hat{\Pi}$; feature function $\phi(s_t)$; and Lagrange multipliers $w$.

Output: A Nash equilibrium ($\hat{\pi}^*, \hat{\pi}^*$).

1: repeat
2: Compute Nash equilibrium ($\hat{\pi}^*_D, \hat{\pi}^*$) and its game value $\hat{v}^*$ for sub-game $\hat{\Pi}$, $\hat{\Pi}$, $\text{loss}(\cdot, \cdot)$, $\phi(\cdot)$, and $w$
3: Compute best response $\hat{\delta}^*$ to $\hat{\pi}^*$ with value $\hat{v}_{\hat{\delta}}$.
4: if $\hat{v}^* \neq \hat{v}_{\hat{\delta}}$ then
5: Add action to set: $\hat{\Pi} \leftarrow \hat{\Pi} \cup \hat{\delta}^*$
6: end if
7: Compute Nash equilibrium ($\hat{\pi}^*, \hat{\pi}^*$) and its game with value $\hat{v}^*$ for sub-game $\hat{\Pi}$, $\hat{\Pi}$, $\text{loss}(\cdot, \cdot)$, $\phi(\cdot)$, and $w$
8: Compute best response $\hat{\delta}^*$ to $\hat{\pi}$ value $\hat{v}_{\hat{\delta}^*}$.
9: if $\hat{v}^* \neq \hat{v}_{\hat{\delta}^*}$ then
10: Add action to set: $\hat{\Pi} \leftarrow \hat{\Pi} \cup \hat{\delta}^*$
11: end if
12: until $\hat{v}_{\hat{\delta}^*} = \hat{v}_{\hat{\delta}^*} = \hat{v}^* = \hat{v}^*$
13: return ($\hat{\pi}^*, \hat{\pi}^*$)
Finding the best response for each player:

\[
\begin{align*}
\argmin_{\hat{\delta}} & \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t) \bigg| \hat{\pi}, \tau, \hat{\pi}, \hat{\tau} \right] ; & \text{or} \\
\argmax_{\hat{\delta}} & \mathbb{E} \left[ \sum_{t=1}^{T} \text{loss}(\hat{S}_t, \hat{S}_t) + w \cdot \phi(\hat{S}_t) \bigg| \hat{\delta}, \tau, \hat{\pi}, \hat{\tau} \right]
\end{align*}
\]

reduces to a time-varying optimal control problem. Consider finding the best demonstrator estimation policy $\hat{\delta}^\ast$. The “expected loss” can be treated as a reward for state $s_t \in S_D$ with a numerical value of:

\[
\text{reward}(s_t) = \mathbb{E}[\text{loss}(s_t, \hat{S}_t)|\hat{\pi}] + w \cdot \phi(s_t).
\]

Once the reward function is constructed, this time-varying optimal control problem is exactly the Finite-Horizon MDP (Section 2.3.2) which can be solved efficiently in $O(|S||A|T)$ time using value iteration (Bellman, 1957). We assume that the set of deterministic policies defining each player’s stochastic policy is relatively small so that marginalizing to compute state rewards is dominated by the run time of solving the optimal control problem. Each player’s best response can be constructed in this manner. Upon termination, neither player’s (mixed) strategy can be improved with an additional game action (i.e., deterministic policy), and, thus, by definition $\hat{\pi}^\ast$ and $\hat{\pi}^\ast$ must be an equilibrium pair (McMahan et al., 2003).

Model parameters $w$ are estimated using a convex optimization routine described in Algorithm 6.2. We refer the reader to Asif et al. (Asif et al., 2015) for the proof of convexity for adversarial prediction learning problems of this form with payoff values that are constant with respect to the probability of each player’s actions, but not the values themselves.
Algorithm 6.2 Learning algorithm for adversarial IOC

**Input:** Demonstration $\tilde{P}(A_{1:T}, S_{1:T})$ from given decision processes, $(\tau, \hat{\tau})$; loss function: $\text{loss}(\cdot, \cdot)$; and learning rate schedule $\lambda_t$

**Output:** Parameters $w$ providing adversarial generalization

1. $w \leftarrow 0$
2. **while** $w$ not converged **do**
3. Compute $\hat{\pi}^*$ from parameters $w$ using double oracle method (Alg. 6.1) given $\tau, \hat{\tau}$
4. Gradient update of parameters: $w \leftarrow w - \lambda_t(E_{P(\hat{S}_{1:T}, \hat{A}_{1:T})}[\sum_{t=1}^{T} \phi(S_t)|\hat{\pi}^*, \tau] - \hat{\epsilon})$
5. **end while**

6.4 Experiments

We demonstrate the benefits of our approach on synthetic and real imitation learning tasks with application-specific imitation losses and/or different embodiments.

6.4.1 Navigation Across a Grid

Our first experiment considers trajectories collected from simulated navigation across a discrete grid with various characteristics. For each task, a robot navigates through the environment to reach a target location. Each cell of the grid world is denoted by its horizontal and vertical positions, $(x, y)$, where each is an integer value from 1 to $N$. The robot’s goal is to reach the target location while minimizing the navigation cost within a fixed period of time. We define this fixed time horizon as the maximum number of steps needed to reach any cell of the grid world. The navigation task stops once the robot reaches the target, which is equivalent to representing that the robot stays in the cell where the target exists until the end of the final time step. We formulate the robot navigation problem to be an optimal sequential decision-making
problem in a finite Markov decision process (MDP) in which the policy minimizes the expected cost of successful navigation.

Differing initial positions for the robot and the target location are sampled uniformly from the $N \times N$ cells. We generate the cost $C(s)$ for the demonstrator to traverse a particular grid cell $(x, y)$ position in the grid) in our simulations based on a linear function of feature vectors, $\phi(s)$, which characterize the state: $C(s) = \theta^T \phi(s) + \varepsilon(s)$, and a noise component, $\varepsilon(s)$. We employ a
7-element feature function vector, $\phi(s)$, in these grid experiments and choose each element of $\theta$ by sampling from the uniform distribution $U(0, 1)$. The noise component is similarly sampled from a uniform distribution, $U(0, \varepsilon)$, bounded by a scalar parameter $\varepsilon$ that controls the amount of noise in the imitation learning task. We set $C(s) = 0$ when the robot reaches the cell where the target exists. Note that the cost is stationary; all values of $C(s)$ are sampled and fixed for each navigation task. The robot can attempt to move one step from its position in each of the cardinal directions (north, south, east and west), except it is unable to move beyond the boundaries of the grid. When the state transition dynamics are stochastic, the robot may accidentally move into another neighboring cell rather the intended one (e.g., north or south when attempting to move east). The state transition dynamics are formally then:

$$p(s_{t+1}|s_t, a_t) = \begin{cases} p_m & \text{matching the action} \\ \frac{1-p_m}{\text{number of neighbor cells}} & \text{neighbor cells} \end{cases}$$

where we call $p_m$ the matching probability. The optimal policy from solving the finite MDP problem gives the robot’s navigation strategy which then can generate a navigation trajectory for learning.

We establish a specific set of grid world navigation simulation characteristics as the base setting of our simulations:

- The size of the grid world is $9 \times 9$;
- The noise weight $\varepsilon$ is 1; and
• The matching probability \( p_m \) is 0.7.

We repeat the simulation 200 times, yielding 200 navigation trajectories of which we use 100 as training data, and the remainder as testing data. We compare adversarial IOC to MMP (Ratliff et al., 2006) (Section 3.2.1) across various settings of the size of the grid, the amount of feature noise, the matching probability, and the number of training/testing datapoints. For our grid navigation experiments, we evaluate the loss as the Euclidean distance between the demonstrator’s grid position \((x, y)\) and the imitator’s grid position \((\hat{x}, \hat{y})\), normalized by the maximum loss, \(m\):

\[
\frac{1}{N_{\text{test}}} \sum_{n=1}^{N_{\text{test}}} \mathbb{E} \left[ \sum_{t=1}^{T} n^{-1} \sqrt{\left( X_t^{(n)} - X_t^{(n)} \right)^2 + \left( Y_t^{(n)} - Y_t^{(n)} \right)^2} \right]
\]

where \((X_t^{(n)} \text{ and } Y_t^{(n)})\) are random variables under the demonstrator’s control policy—the policy from solving the simulated finite MDP problem—and \((\hat{X}_t^{(n)} \text{ and } \hat{Y}_t^{(n)})\) are the ones with estimated policy. We employ this normalized Euclidean loss as the structured loss function for the margin in MMP and the game payoff in our adversarial method.

As shown in the first four plots of Figure 6.2, our adversarial IOC approach (Adv) provides significant improvements in reducing the imitation loss over the trajectory compared to maximum margin planning (MMP) under equivalent embodiment setting (i.e., standard imitation learning). Though the imitator’s performance generally becomes worse as the imitation task becomes more difficult (less determinism in the state transition dynamics, increased amounts of noise influencing the demonstrator’s optimal policy, and larger sizes of the grid), adversarial IOC consistently outperforms MMP across all of these settings. Very little dependence of the
imitation performance on the number of training examples in the fourth plot reveals the general
efficiency of training using IOC/IRL methods that estimate the motivating cost function.

We also compare the performance of our adversarial IOC imitation policy with the policy
produced by MMP (Ratliff et al., 2006) when demonstrator and imitator have different embodi-
ments. We assume that the demonstration robot’s dynamics are noise free and more flexible. In
our first experiment, the demonstrator has deterministic state transition dynamics with match-
ing probability 1, and we evaluate the performance of the learner operating under stochastic
dynamics with various matching probabilities from 0.9 to 0.5. In the second experiment, we
set some obstacles in the grid world so that the imitating robot has to make a detour when it
faces any of them, but the demonstrator does not. We evaluate the performance of learner on
various number of obstacles from 20 to 60.

The performance of the two methods under different embodiments is similarly evaluated
according to the average expected trajectory loss of withheld test data, as shown in the final two
plots of Figure 6.2. Our adversarial IOC method also outperforms MMP in these experimental
settings.

6.4.2 Learning Camera Control from Demonstration

We consider the task of learning to autonomously control a camera in a manner that appro-
priately captures the action of a basketball game based on human demonstrations of camera
control (Chen and Carr, 2015). The decision process characterizing camera control can be di-
vided into a probabilistic model describing the state of the basketball game (the presence of
players in different locations), and a dynamics model describing how camera movement controls
effect the camera’s state (quantized pan angle, $\theta$, and quantize pan angle velocity, $\dot{\theta}$). As our focus is on the separation of rationalization and imitation evaluation measure, we assume that camera controls have no influence on the basketball game. Also based on this focus, we employ the empirical distribution of player locations rather than constructing a predictive model for those locations.

Our dataset is collected from high school basketball games. The camera recording the basketball game was operated by a human expert. The dataset consists of 46 sequences collected at 60Hz. The average number of frames for the sequences is 376. The output for each frame is the camera’s horizontal pan angle, and the input is a 14 element vector that describes the state of the basketball game (the presence of players in different locations on the basketball court). The degree of the camera’s pan angle in this dataset ranges from $-30$ degrees (left) to 30 degree.
(right), and we quantize the pan angle $\theta$ into discrete 61 levels. The pan angel velocity $\dot{\theta}$ of a particular frame is the difference between the current pan angle and the previous one, which is then mapped to 5 discrete levels $[-2, -1, 0, 1, 2]$ representing high speed of turning left to high speed of turning right. Overall, by combining the discrete pan angles and pan angle velocities, there are 305 total possible states for each frame. We use the first 23 sequences as our training dataset and the 23 remaining sequences as the testing dataset. We measure the performance of our adversarial IOC method and baseline methods using the average square loss per frame between pan angles:

$$
\sum_{n=1}^{N} \sum_{t=1}^{T_n} (\theta_{t}^{(n)} - \hat{\theta}_{t}^{(n)})^2 / \sum_{n=1}^{N} T_n.
$$

(6.4.1)

We compare our adversarial structured prediction method with a few forms of least squares linear regression models: one that is not constrained by the camera dynamics (LS); one that is constrained by the empirical dynamics of the camera (LS$_C$); and one Markovian-based model that also conditions on the previous camera location (LS$_M$). Additionally, we consider two variants of maximum marginal planning methods: MMP$_{SL}$ is provided with the starting location of the human-operated camera, while MMP is not. Similarly, Adv$_{SL}$ is our adversarial IOC method provided with the starting location of the human-operated camera, while Adv is not.

Let $X_t$ denotes the 14 entry feature vector of the state of the basketball game at timestep $t$. The feature vector $\phi(S_t)$ of our adversarial method in definition 6.3 is a 33 entry vector $[\theta, \theta^2, \dot{\theta}, \dot{\theta}^2, \theta X_t, \dot{\theta} X_t]$, which combines the basketball game state features and the camera angle
and angle velocity state. For the regression models, the estimated sequence is a standard linear regression method \( \hat{\theta}_t = \hat{a}X_t + \hat{b} \) where \( \hat{a} \) and \( \hat{b} \) are trained from the training dataset. For the constrained regression method, the predicted camera angle is projected to the closest angle for which transitioning is feasible.

The result of a test sequence of our experiment is shown in the left plot of Figure 6.3. The first two regression methods are generally very noisy as the predicted pan angle changes rapidly based on the rapid changes of the underlying inputs corresponding to the game state. The Markovian regression model performs well initially, but diverges from the demonstrated trajectory over time. Both of the MMP methods have much worse performance than the other methods presented. Our adversarial approaches tend to be similar to the regression model, but are much less noisy and provide a closer match to the demonstrated trajectory with significantly lower amounts of squared and absolute loss, as shown in the other plots of Figure 6.3.

6.5 Summary

In this chapter, we introduced an adversarial framework for imitation learning using inverse optimal control. It takes the form of a game between an adversary seeking to maximize loss by approximating the training data, and a learner seeking to minimize the loss. A key benefit of our approach is that it separates the rationalization of demonstrated decision sequences with the learner’s optimization of an imitative loss function. We focused this added flexibility on the problem of learning to imitate under differences in embodiment. This is an underexplored, but important problem for imitation learning to be employed in practice. We established the consistency and useful generalization bounds for our adversarial inverse optimal control
approach. We developed and presented efficient algorithms for inference and learning under this formulation. Finally, we demonstrated the benefits of adversarial inverse optimal control in a set of synthetic experiments and an autonomous camera control task where an autonomous camera is trained based on observations of human camera control.
CHAPTER 7

CONCLUSION

7.1 Discussion

This thesis introduced our work on developing robust structured prediction models for process data. We demonstrated the benefits and effectiveness of incorporating partial observability of our approach based on the principle of maximum causal entropy, for a synthetically created control policy estimation task and for modeling mouse cursor pointing motions in chapter 4. In chapter 5 we introduced our work on dealing with covariate shift for linear regression. Our approach motivated by a minimax estimation formulation is reduced to minimizing a conditional Kullback-Leibler divergence subject to matching statistics. We demonstrated the robust prediction performance of our approach on both synthetically created and natural existing covariate shift settings of a range of real regression datasets including large size and high dimensional datasets. In chapter 6 we showed the benefits of our adversarial approach which provides a minimax estimation framework for enabling general imitative losses and the consideration of differences in capabilities between the demonstrator and the learner, on a synthetically created grid word navigation and an autonomous camera control learning tasks.

As we showed in section 4.3.2 and section 6.4.2 linear regression (Section 3.1.1), a direct estimation approach for continuous input and output setting, has much worse performance than our robust approaches in modeling mouse cursor motion as well as learning camera control. This
is because direct estimation (Section 3.1) is not adaptive to process prediction tasks especially ones under decision control settings, and it also doesn’t consider learning process prediction models in its entirety.

Early work of inverse reinforcement learning (IRL) (Section 3.2) attempts to recover a reward function from training samples assuming optimal process strategies are generalized in these samples so as to learn a policy using reinforcement learning techniques. Maximum marginal approach (MMP) (Section 3.2.1) is proposed to solve the ambiguity by seeking a reward function that makes process strategies generalized in training samples outperforms all alternatives by a structure loss measure and also maintains the maximum margin. But, as we showed in section 6.4, the prediction performance of MMP is unstable and worse than our robust approach on grid world navigation and learning camera control tasks. This is because the optimality assumption early IRL methods assume often doesn’t hold in real process prediction tasks. The prediction performance of these early IRL methods is very sensitive to imperfect training samples.

Process environments are inherently complex and noisy. Our conclusion by summarizing the theoretical and empirical argument in this thesis is that process prediction models that are developed by deriving the best probability distribution estimation under the worst case subject to matching known properties of the real distribution demonstrate robust prediction performance and will benefit from incorporating partial observability, dealing with non-stationary process settings and enabling various evaluation measures and embodiment transfer.
7.2 Future Work

Throughout this thesis, we have seen that the minimax approach achieves robust prediction performance in various process settings. However, extending the minimax approach to general process settings with competitive prediction performance and simultaneously achieving computationally efficient learning and prediction is still challenging, leading to a list of open research questions.

The assumption that the process setting in which training samples are collected and the one in which the learned model will be deployed share the same property often doesn’t hold in practice. Instead, learning a prediction model for process data is often needed under a non-stationary environment. One form of this that we investigated in chapter \[i\] is known as covariate shift. We demonstrated the benefit of our robust approach in dealing with covariate shift for linear regression. We believe that extending our approach to deal with covariate shift for general process prediction problems will achieve great prediction improvement in real process prediction tasks. For instance, we can incorporate maximum causal entropy, a structured prediction model evaluated using log-loss over the entire process, with our robust approach to deal with covariate shift for process prediction problems.

As for partial observability, we showed the benefit of incorporating partial observability in modeling mouse cursor motion in our predictive inverse optimal control for the LQG work. However, our method for dealing with partial observability takes the benefit of the linearity property of the dynamics and the Gaussian distribution assumption and is restricted to adversarially min-
imizing log-loss. Extending the minimax approach to incorporate partial observability (Choi and Kim, 2011) for general process prediction problems is worth investigating.

Our adversarial imitation learning framework enables learning for general evaluation measures and different capabilities between the learner and demonstrator. These benefits make it potentially an important learning model for general process prediction problems. However, our framework hasn’t taken into account how to deal with large number of states and actions (the scalability issue). Scalability is also a challenging issue for many other learning methods such as reinforcement learning (Sutton and Barto, 1998). Some useful methods have been proposed including heuristic state space search (Monfort et al., 2015) and value function approximation (Sutton, 1988; Werbos, 1990; Bertsekas and Tsitsiklis, 1995). Incorporating these methods into our adversarial imitation learning framework to improve scalability deserves exploration.

Features, as measurable properties of a phenomenon being observed, play an important role in learning. Our work throughout this thesis focuses on developing robust structured prediction model for process data under various process settings without taking into account feature construction. Important future work could construct more representative feature functions for robust structured prediction model such as kernelization or embedding a deep neural network (Finn et al., 2016), which may greatly improve the prediction performance of the model.
APPENDICES
Appendix A

PROOFS

A.1 Chapter 4 Proofs

Lemma 5. The constrained optimization of equation 4.2.9 is equivalent to:

\[
\begin{align*}
\arg \max_{\{f(u_{1:T}||z_{1:T},x_{1:T})\}} & H(U_{1:T}||Z_{1:T}, X_{1:T}) \\
\text{where: } f(u_{1:t}|z_{1:T}, x_{1:T}) & = \prod_{t=1}^{T} f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}); \\
\forall t \in \{1 \cdots T\}, u_{1:t} \in U_{1:t}, z_{1:t} \in Z_{1:t}, x_{1:t} \in X_{1:t}, x'_{1:t} \in X_{1:t}, \\
such that f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) & \geq 0, \int_{u_t \in U_t} f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) = 1,
\end{align*}
\]

\[f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) = f(u_t|u_{1:t-1}, z_{1:t}, x'_{1:t}).\] (A.1.4)

Proof of Lemma 5. The previously developed theory of maximum causal entropy (Ziebart et al., 2010) shows the causally conditioned probability distribution defined according to affine constraint (Equation 4.2.6, 4.2.8) are equivalent to it defined by the decomposition into a product of conditional probabilities (Equation A.1.2, A.1.3). Then, we show the partial observability constraint (Equation 4.2.7) implies equation A.1.4.

\[
\forall u_{1:T} \in U_{1:T}, x_{1:T}, x'_{1:T} \in X_{1:T}, z_{1:T} \in Z_{1:T},
\prod_{t=1}^{T} f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) = \prod_{t=1}^{T} f(u_t|u_{1:t-1}, z_{1:t}, x'_{1:t})
\]
Appendix A (Continued)

It is possible, \( f(u_1|z_1, x_1) \cdots f(u_T|u_{1:T-1}, z_{1:T}, x_{1:T}) \)

\[ = f(u_1|z_1, x_1) \cdots f(u_t|u_{1:t-1}, z_{1:t}, x'_{1:t}) \cdots f(u_T|u_{1:T-1}, z_{1:T}, x_{1:T}) \]

Thus, \( f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) = f(u_t|u_{1:t-1}, z_{1:t}, x'_{1:t}) \).

It is easy to show equation A.1.4 implies the partial observability constraint (Equation 4.2.7).

\[ \square \]

**Lemma 6.** The constrained optimization defined in Lemma 5 is equivalent to:

\[
\arg \max \{f(u_1:T||z_1:T)\} H(U_1:T||Z_{1:T}) \tag{A.1.5}
\]

\[ \forall u_{1:T} \in U_{1:T}, z_{1:T}, z'_{1:T} \in Z_{1:T}, \]

\[ f(u_{1:T}||z_{1:T}) \geq 0, \int_{u'_{1:T} \in U_{1:T}} f(u'_{1:T}||z_{1:T}) \, du'_{1:T} = 1, \tag{A.1.6} \]

\[ \forall \tau \in \{ 1, \ldots, T \} \text{ such that } z_{1:\tau} = z'_{1:\tau}, \]

\[ \int_{u_{\tau+1:T} \in U_{\tau+1:T}} f(u_{1:T}||z_{1:T}) \, du_{\tau+1:T} = \int_{u_{\tau+1:T} \in U_{\tau+1:T}} f(u_{1:T}||z'_{1:T}) \, du_{\tau+1:T}. \tag{A.1.7} \]

**Proof of Lemma 6**

\[ \forall t \in \{ 1, \ldots, T \}, u_{1:t} \in U_{1:t}, z_{1:t} \in Z_{1:t}, x_{1:t} \in X_{1:t}, x'_{1:t} \in X_{1:t}, \]

\[ f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) = f(u_t|u_{1:t-1}, z_{1:t}, x'_{1:t}) = f(u_t|u_{1:t-1}, z_{1:t}) \]

Then, \( \prod_{t=1}^{T} f(u_t|u_{1:t-1}, z_{1:t}, x_{1:t}) = \prod_{t=1}^{T} f(u_t|u_{1:t-1}, z_{1:t}) = f(u_{1:T}||z_{1:T}) \)
Appendix A (Continued)

Similar to the proof of Lemma 5, the causally conditioned probability distribution defined by
a product of conditional probabilities are equivalent to the affine constraint (Equation A.1.6
A.1.7).

To show the object function (Equation A.1.5) is equivalent to equation 4.2.9, we first show

\[
\int_{x_1:T \in X_1:T} f(z_1:T, x_1:T | u_1:T-1) \\
= \int_{x_1:T \in X_1:T} \prod_{t=1}^{T} f(z_t, x_t | z_{1:t-1}, x_{1:t-1}, u_{1:t-1}) \prod_{t=1}^{T} f(u_t | u_{1:t-1}, z_{1:t}) \\
= \int_{x_1:T \in X_1:T} \prod_{t=1}^{T} f(u_t | u_{1:t-1}, z_{1:t}) f(z_t | z_{1:t-1}, x_{1:t}) \\
= \prod_{t=1}^{T} f(u_t | u_{1:t-1}, z_{1:t}) \prod_{t=1}^{T} f(z_t | z_{1:t-1}, u_{1:t-1}) \\
= \prod_{t=1}^{T} f(z_t | z_{1:t-1}, u_{1:t-1}) = f(z_1:T | u_1:T-1)
\]

Then, \(H(U_1:T | Z_1:T, X_1:T)\)

\[
H(U_1:T | Z_1:T, X_1:T) \\
= \int_{u_1:T, z_1:T, x_1:T} f(u_1:T | z_1:T, x_1:T) f(z_1:T, x_1:T | u_1:T-1) \log f(u_1:T | z_1:T, x_1:T) \\
= \int_{u_1:T, z_1:T} f(u_1:T | z_1:T) \int_{z_1:T} f(z_1:T, x_1:T | u_1:T-1) \log f(u_1:T | z_1:T, x_1:T) \\
= \int_{u_1:T, z_1:T} f(u_1:T | z_1:T) \log f(u_1:T | z_1:T) f(z_1:T | u_1:T-1) \\
= H(U_1:T | Z_1:T)
\]
Lemma 7. Suppose the constrained optimization problem in Lemma 6 has the following additional constraint: \( F : \mathcal{U}_{1:T} \times \mathcal{Z}_{1:T} \rightarrow \mathbb{R}^N, c \in \mathbb{R}^N \)

\[
\mathbb{E}_{f(u_{1:T},z_{1:T})}[F(U_{1:T}, Z_{1:T})] = c \tag{A.1.8}
\]

Then the solution to this optimization problem has the form:

\[
f_t(u_t|u_{1:t-1}, z_{1:t}) = e^{Q(u_{1:t}, z_{1:t}) - V(u_{1:t-1}, z_{1:t})}
\]

where \( Q \) and \( V \) functions take the following recursive form:

\[
Q(u_{1:t}, z_{1:t}) = \begin{cases} 
\lambda^T F(u_{1:T}, z_{1:T}), & t = T; \\
\mathbb{E}[V(U_{1:t}, Z_{1:t+1})|u_{1:t}, z_{1:t}], & t < T
\end{cases}
\]

\[
V(u_{1:t-1}, z_{1:t}) = \text{softmax} Q(u_{1:t}, z_{1:t}) \triangleq \log \int_{u_t} e^{Q(u_{1:t}, z_{1:t})} du_t
\]

Proof of Lemma 7. We first show for any joint distribution \( g(u_{1:T}, z_{1:T}) \), the following equation holds:

\[
\mathbb{E}_g \left[ - \log \hat{f}(U_{1:T}|Z_{1:T}) \right] = \int_{z_{1:T}} f(z_{1:T})V(z_{1:T}) - \mathbb{E}_g \left[ \lambda^T F(U_{1:T}, Z_{1:T}) \right] \tag{A.1.9}
\]
Appendix A (Continued)

\[
\mathbb{E}_g \left[ \sum_{t=1}^{T} - \log \hat{f}(U_t|U_{1:t-1}, Z_{1:t}) \right] \\
= \mathbb{E}_g \left[ -\lambda^T F(U_{1:T}, Z_{1:T}) - \sum_{l=1}^{T-1} Q(U_{1:t}, Z_{1:t}) + \sum_{l=1}^{T} V(U_{1:t-1}, Z_{1:t}) \right] \\
= \mathbb{E}_g \left[ -\lambda^T F(U_{1:T}, Z_{1:T}) \right] - \int g(u_{1:T}, z_{1:T}) \sum_{l=1}^{T-1} \int f(z_{l+1}|u_{1:t}, z_{1:t}) V(u_{1:t}, z_{1:t+1}) \\
+ \int g(u_{1:T}, z_{1:T}) \sum_{l=1}^{T} V(u_{1:t-1}, z_{1:t}) \\
= \mathbb{E}_g \left[ -\lambda^T F(U_{1:T}, Z_{1:T}) \right] - \sum_{l=1}^{T-1} \int g(u_{1:T}, z_{1:t+1}) V(u_{1:t}, z_{1:t+1}) \\
+ \int g(u_{1:T}, z_{1:T}) \sum_{l=1}^{T} V(u_{1:t-1}, z_{1:t})
\]

For any arbitrary causally conditional probability distribution \(g(u_{1:T}|z_{1:T})\) satisfies with expectation constraint (Equation [A.1.8]), we show:

\[
H_g(U_{1:T}||Z_{1:T}) \leq H_f(U_{1:T}||Z_{1:T})
\]

\[
\mathbb{E}_g \left[ -\log g(U_{1:T}|Z_{1:T}) \right] \\
= - \int g(u_{1:T}, z_{1:T}) \log \left( \frac{g(u_{1:T}|z_{1:T}) f(z_{1:T}|u_{1:T-1})}{\hat{f}(u_{1:T}|z_{1:T}) f(z_{1:T}|u_{1:T-1})} \right) \\
= - D_{KL} \left( g(u_{1:T}, z_{1:T}) || \hat{f}(u_{1:T}, z_{1:T}) \right) - \int g(u_{1:T}, z_{1:T}) \log \hat{f}(u_{1:T}|z_{1:T}) \\
\leq - \int g(u_{1:T}, z_{1:T}) \log \hat{f}(u_{1:T}|z_{1:T}) = \int f(z_{1}) V(z_{1}) - \mathbb{E}_g [\lambda^T F(U_{1:T}, Z_{1:T})] \\
= \int f(z_{1}) V(z_{1}) - \mathbb{E}_f [\lambda^T F(U_{1:T}, Z_{1:T})] = H_f(U_{1:T}|Z_{1:T})
\]
Appendix A (Continued)

$D_{KL}$ is the Kullback-Leibler divergence which is non-negative (Cover and Thomas, 2006). Thus, $\hat{f}(u_t|u_{1:t-1}, z_{1:t})$ is the solution to the optimization problem in Lemma 6 incorporating with expectation constraint (Equation A.1.8).

Proof of Theorem 4.1. We first incorporate the expectation constraint (Equation 4.2.10) into the constrained optimization problem defined in Lemma 6

\[
\mathbb{E}_{f(u_{1:T}, z_{1:T+1}, x_{1:T+1})} \left[ \sum_{t=1}^{T+1} X_t x_t^T \right] \\
= \int_{u_{1:T}, z_{1:T+1}, x_{1:T+1}} f(u_{1:T}|z_{1:T}, x_{1:T}) f(z_{1:T+1}, x_{1:T+1}|u_{1:T}) \sum_{t=1}^{T+1} x_t x_t^T \\
= \int_{u_{1:T}, z_{1:T}} f(u_{1:T}|z_{1:T}) f(z_{1:T}|u_{1:T-1}) \frac{\int_{x_{1:T+1}, z_{T+1}} f(z_{1:T+1}, x_{1:T+1}|u_{1:T}) \sum_{t=1}^{T+1} x_t x_t^T}{f(z_{1:T}|u_{1:T-1})} \\
= \mathbb{E}_{f(u_{1:T}, z_{1:T})} \left[ \sum_{t=1}^{T+1} x_t x_t^T \right]
\]

According to Lemma 7, the solution to the constrained problem defined in Lemma 6 incorporates with the expected constraint (Equation A.1.8) takes the following recursive form:

\[
Q(u_{1:t}, z_{1:t}) = \begin{cases} 
\frac{\int_{x_{1:T+1}, z_{T+1}} f(z_{1:T+1}, x_{1:T+1}|u_{1:T}) \sum_{t=1}^{T+1} x_t x_t^T}{f(z_{1:T}|u_{1:T-1})}, & t = T; \\
\mathbb{E}[V(U_{1:t}, Z_{1:t+1})|u_{1:t}, z_{1:t}], & t < T
\end{cases}
\]

\[
V(u_{1:t-1}, z_{1:t}) = \text{softmax}_{u_t} Q(u_{1:t}, z_{1:t}) \triangleq \log \int_{u_t} e^{Q(u_{1:t}, z_{1:t})} du_t
\]
Appendix A (Continued)

\[ Q(u_{1:T}, z_{1:T}) = \frac{\int x_{1:T} f(z_{1:T}, x_{1:T} | u_{1:T-1}) \mathbb{E} \left[ X_{T+1}^T M X_{T+1} | X_T, u_T \right]}{f(z_{1:T} | u_{1:T-1})} \]

We define it \( Q'(u_{1:T}, z_{1:T}) \)

\[ + \frac{\int x_{1:T} f(z_{1:T}, x_{1:T} | u_{1:T-1}) \sum_{t=1}^T x_t^T M x_t}{f(z_{1:T} | u_{1:T-1})} \]

This is a constant term with respect to \( u_T \), we define it \( W_T \)

\[ Q'(u_{1:T}, z_{1:T}) = \frac{\int x_{1:T} f(z_{1:T}, x_{1:T} | u_{1:T-1}) f(u_{1:T} | x_{1:T}, z_{1:T}) \mathbb{E} \left[ X_{T+1}^T M X_{T+1} | X_T, u_T \right]}{f(z_{1:T} | u_{1:T-1}) f(u_{1:T} | z_{1:T})} \]

\[ = \frac{\int x_{T+1} f(x_{T+1}, u_{1:T}, z_{1:T}) x_{T+1}^T M x_{T+1}}{f(u_{1:T}, z_{1:T})} \]

\[ = \mathbb{E} \left[ X_{T+1}^T M X_{T+1} | u_{1:T}, z_{1:T} \right] \]

Let \( V'(u_{1:T-1}, z_{1:T}) = \log \int_{u_T} Q'(u_{1:T}, z_{1:T}) \)

\[ Q(u_{1:T-1}, z_{1:T-1}) = \int_{z_{T}} f(z_T | u_{1:T-1}, z_{1:T-1}) (W_T + V'(u_{1:T-1}, z_{1:T})) \]

\[ = \frac{\int x_{T} f(z_{1:T}, x_{1:T} | u_{1:T-1}) f(u_{1:T-1} | x_{1:T-1}, z_{1:T-1}) x_{T}^T M x_{T}}{f(z_{1:T-1} | u_{1:T-2}) f(u_{1:T-1} | z_{1:T-1})} \]

\[ + \mathbb{E} \left[ V'(U_{1:T-1}, Z_{1:T}) | u_{1:T-1}, z_{1:T-1} \right] + W_{T-1} \]

\[ = \mathbb{E} \left[ X_{T}^T M X_{T} + V'(U_{1:T-1}, Z_{1:T}) | u_{1:T-1}, z_{1:T-1} \right] + W_{T-1} \]

We define it \( Q'(u_{1:T-1}, z_{1:T-1}) \)

And let \( V'(u_{1:T-2}, z_{1:T-1}) = \log \int_{u_{T-1}} Q'(u_{1:T-1}, z_{1:T-1}) \)

For \( t < T-1 \), the argument to \( Q'(u_{1:t}, z_{1:t}) \), \( V'(u_{1:t-1}, z_{1:t}) \) is similar. We redefine \( Q(u_{1:t}, z_{1:t}) = Q'(u_{1:t}, z_{1:t}) \) and \( V(u_{1:t-1}, z_{1:t}) = V'(u_{1:t-1}, z_{1:t}) \) which gives the recursive form in Theorem 4.1.
Lemma 8. The distribution of belief state $X_t|b_t \sim N(\mu_{b_t}, \Sigma_{b_t})$ is recursively defined as following and $\Sigma_{b_t}$ is independent of $b_t$.

$$
\mu_{b_1} = \mu + \Sigma_{d_1}^T C^T (\Sigma_o + C \Sigma_{d_1}^T C^T)^{-1} (Z_1 - C \mu) \tag{A.1.10}
$$

$$
\Sigma_{b_1} = \Sigma_{d_1} - \Sigma_{d_1}^T C^T (\Sigma_o + C \Sigma_{d_1}^T C^T)^{-1} C \Sigma_{d_1} \tag{A.1.11}
$$

$$
\mu_{b_{t+1}} = BU_t + A \mu_{b_t} + (\Sigma_d + A \Sigma_{b_t}^T A^T)^T C^T
\begin{equation}
(\Sigma_o + C (\Sigma_d + A \Sigma_{b_t}^T A^T)^T C^T)^{-1} (Z_{t+1} - C (BU_t + A \mu_{b_t})) \tag{A.1.12}
\end{equation}
$$

$$
\Sigma_{b_{t+1}} = \Sigma_d + A \Sigma_{b_t}^T A^T - (\Sigma_d + A \Sigma_{b_t}^T A^T)^T C^T
\begin{equation}
(\Sigma_o + C (\Sigma_d + A \Sigma_{b_t}^T A^T)^T C^T)^{-1} C (\Sigma_d + A \Sigma_{b_t}^T A^T) \tag{A.1.13}
\end{equation}
$$

Proof of Lemma Since $Z_1|x_1 \sim N(Cx_1, \Sigma_o)$ and $X_1 \sim N(\mu, \Sigma_d)$, applying Gaussian transformation techniques, it is easy to show that the distribution of initial belief state $X_1|b_1$ (that is $X_1|z_1$) is a Gaussian distribution with mean (Equation A.1.10) and variance (Equation A.1.11).

Note that $f(x_{t+1}|x_t, u_t, b_t) = f(x_{t+1}|x_t, u_t)$ $X_{t+1}|x_t, u_t \sim N(Ax_t + Bu_t, \Sigma_d)$

$$
f(x_t|u_t, b_t) = f(x_t|b_t)$ $X_t|b_t \sim N(\mu_b, \Sigma_{b_t})
$$

Then $X_{t+1}|u_t, b_t \sim N(Bu_t + A \mu_{b_t}, \Sigma_d + A \Sigma_{b_t}^T A^T)$

Furthermore $f(z_{t+1}|x_{t+1}, u_t, b_t) = f(z_{t+1}|x_{t+1})$ $Z_{t+1}|x_{t+1} \sim N(Cx_{t+1}, \Sigma_o)$
Thus, it’s easy to show the distribution of $X_{t+1}, Z_{t+1}|u_t, b_t$ is:

$$
\mathbb{N} \left( Bu_t + A\mu_{b_t}, \Sigma_d + A\Sigma_{b_t}A^T, (\Sigma_d + A\Sigma_{b_t}A^T)^T C^T \right).
$$

Finally, $f(x_{t+1}|b_{t+1}) = f(x_{t+1}, z_{t+1}|u_t, b_t)/f(z_{t+1}|u_t, b_t)$ which gives the distribution of $X_{t+1}|b_{t+1}$ with mean (Equation A.1.12) and variance (Equation A.1.13).

**Proof of Theorem 4.2.**

$$
\mathbb{E}[X_{t+1}^T M X_{t+1}|u_{1:t}, z_{1:t}] = \mathbb{E}[X_{t+1}^T M X_{t+1}|u_t, b_t]
$$

$$
= (Bu_t + A\mu_{b_t})^T M(Bu_t + A\mu_{b_t}) + tr(M(\Sigma_d + A\Sigma_{b_t}A^T)) + \text{constant}
$$

Thus $Q(u_{1:T}, z_{1:T}) = Q(u_T, \mu_{b_T}) = \mathbb{E}[X_{t+1}^T M X_{t+1}|u_t, b_t]$ gives $W_T$. $V(u_{1:T-1}, z_{1:T}) = V(\mu_{b_T}) = V(z_T, u_{T-1}, \mu_{T-1}) = \log \int_{u_T} e^{Q(u_T, \mu_{b_T})}$

$$
= \mu_{b_T}^T (W_{T(\mu, \mu)} - W_{T(U, \mu)}^T W_{T(U, U)}^{-1} W_{T(U, \mu)}) \mu_{b_T} + \text{constant}
$$

$$
= \begin{bmatrix} z_T \\ u_{T-1} \\ \mu_{b_{T-1}} \end{bmatrix}^T \begin{bmatrix} P_T^T (W_{T(\mu, \mu)} - W_{T(U, \mu)}^T W_{T(U, U)}^{-1} W_{T(U, \mu)}) P_T \\ \mu_{b_{T-1}} \end{bmatrix} + \text{constant}
$$

which gives $D_T$. 

Thus $E[V(U_{1:t}, Z_{1:t+1})|u_{1:t}, z_{1:t}] = E[V(Z_{t+1}, U_t, \mu_{b_t})|u_t, \mu_{b_t}]$

\[
= E \left[ \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix}^T D_{t+1(u_{z_t}, z_{1:t+1})} + Z_{t+1}^T D_{t+1(z, u_{1:t})} \begin{bmatrix} u_{1:t} \\ \mu_{b_t} \end{bmatrix} \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix} \right] + \text{constant}
\]

\[
+ E \left[ Z_{t+1}^T D_{t+1(z, z_{1:t})} \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix} \right] + \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix} D_{t+1(u_{1:t}, u_{1:t})} \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix}
\]

\[
= \left[ \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix}^T D_{t+1(u_{1:t}, u_{1:t})} C_{BA} + C_{BA}^T D_{t+1(z, u_{1:t})} + C_{BA}^T D_{t+1(u_{1:t}, u_{1:t})} C_{BA} + D_{t+1(u_{1:t}, u_{1:t})} \begin{bmatrix} u_t \\ \mu_{b_t} \end{bmatrix} \right]
\]

$Q(u_t, \mu_{b_t}) = E[X_{t+1}^T M X_{t+1} + V(Z_{t+1}, U_t, \mu_{b_t})|u_{1:t}, z_{1:t}]$ which gives $W_t$. The quadratic form of $V(z_t, u_{t-1}, \mu_{t-1})$ is similar to $V(z_T, u_{T-1}, \mu_{T-1})$ which gives $D_t$. \hfill \Box

**Proof of Theorem 4.3** It’s easy to check the initial setting $W_T = \begin{bmatrix} B & A \end{bmatrix}^T M \begin{bmatrix} B & A \end{bmatrix}$ matches equation 4.1.12. For general case, we plug $D_{t+1}$ into $W_t$. To simplify proof, let’s define

\[
\phi_t = W_t(u_{1:t}, \mu_{1:t}) - W_{t(U_t, \mu_t)}^{-1} W_{t(U_t, \mu_t)} W_t(U_{1:t}, \mu_{1:t})
\]

\[
W_t = \left[ \begin{bmatrix} B & A \end{bmatrix}^T M \begin{bmatrix} B & A \end{bmatrix} + \begin{bmatrix} B - E_{t+1} C B & A - E_{t+1} C A \end{bmatrix}^T \phi_{t+1} E_{t+1} \begin{bmatrix} C B & C A \end{bmatrix} + \right.
\]

\[
\left. \begin{bmatrix} C B & C A \end{bmatrix}^T E_{t+1} \phi_{t+1} E_{t+1} \begin{bmatrix} B - E_{t+1} C B & A - E_{t+1} C A \end{bmatrix} + \right.
\]

\[
\left. \begin{bmatrix} C B & C A \end{bmatrix}^T E_{t+1} \phi_{t+1} E_{t+1} \begin{bmatrix} C B & C A \end{bmatrix} + \right.
\]

\[
\left. \begin{bmatrix} B - E_{t+1} C B & A - E_{t+1} C A \end{bmatrix}^T \phi_{t+1} E_{t+1} \begin{bmatrix} B - E_{t+1} C B & A - E_{t+1} C A \end{bmatrix} \right]
\]
Appendix A (Continued)

\[ W_{t(U,U)} = B^T MB + (B - E_{t+1}CB)^T \phi_{t+1}E_{t+1}CB + (E_{t+1}CB)^T \phi_{t+1}(B - E_{t+1}CB) + (E_{t+1}CB)^T \phi_{t+1}E_{t+1}CB + (B - E_{t+1}CB)^T \phi_{t+1}(B - E_{t+1}CB) \]

\[ = B^T MB + B^T \phi_{t+1}B \]

That is \( B^T F_{t+1}B = B^T MB + B^T \phi_{t+1}B \). \hspace{1cm} (A.1.14)

By plugging out \( \phi_{t+1} \), equation A.1.14 matches equation 4.1.12. \( W_{t(U,\mu)}, W_{t(\mu,U)}, W_{t(\mu,\mu)} \) follow a similar argument. \( \square \)
Appendix A (Continued)

A.2 Chapter 5 Proofs

Proof of Theorem 5.1. With strong duality holds,

\[
\min_{f(y|x) \in \Delta} \max_{f(y|x) \in \Delta \cap \Xi} \text{rel-loss}_{f_{trg}(x)} \left( f(Y|X), \hat{f}(Y|X), f_0(Y|X) \right)
\]

\[
= \max_{f(y|x) \in \Delta \cap \Xi} \min_{f(y|x) \in \Delta} \text{rel-loss}_{f_{trg}(x)} \left( f(Y|X), \hat{f}(Y|X), f_0(Y|X) \right)
\]

\[
\left( \text{Given } f(y|x), \text{ the minimum of relative logloss is achieved when } \hat{f}(y|x) = f(y|x) \right)
\]

\[
= \max_{f(y|x) \in \Delta \cap \Xi} \text{rel-loss}_{f_{trg}(x)} \left( f(Y|X), f(Y|X), f_0(Y|X) \right)
\]

\[
= \min_{f(y|x) \in \Delta \cap \Xi} D_{f_{trg}(x),f(y|x)} \left( f(Y|X)||f_0(Y|X) \right)
\]

\[
= \min_{f(y|x) \in \Delta \cap \Xi} D_{f_{trg}(x),f(y|x)} \left( \hat{f}(Y|X)||f_0(Y|X) \right)
\]

\[
\square
\]

Proof of Theorem 5.2. Let

\[
\hat{f}_\theta(y|x) = f_{o}(y|x)e^{-\frac{f_{src}(x)\theta^T \Phi(x,y)}{Z(x,\theta)}},
\]

where \(Z(x,\theta)\) is the normalization term which guarantees \(\hat{f}_\theta(y|x)\) is a valid conditional probability, and \(\theta\) is chosen so that \(\hat{f}_\theta(y|x)\) satisfies statistic constraint. Formally, \(\hat{f}_\theta(y|x) \in \Delta \cap \Xi\).

Then \(\hat{f}_\theta(y|x)\) uniquely minimizes the Kullback-Leibler divergence over all conditional probabilities \(g(y|x) \in \Delta \cap \Xi\).
Appendix A (Continued)

To simplify the proof let

$$\phi(x, y) = -\frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} \theta^T \Phi(x, y) - \log Z(x, \theta),$$

and then

$$D(g(Y|X)||f_o(Y|X))$$

$$= \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log \frac{g(y|x)}{f_o(y|x)}$$

$$= \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log \frac{g(y|x)}{\hat{f}(y|x)} \hat{f}(y|x) - \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log f_o(y|x)$$

$$= D(g(Y|X)||\hat{f}_\theta(Y|X)) + \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log \hat{f}(y|x) - \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log f_o(y|x)$$

$$\geq \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log \hat{f}(y|x) - \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \log f_o(y|x)$$

$$= \int_X f_{\text{trg}}(x) \int_{Y|X} g(y|x) \phi(x, y) \quad \left(\text{Both } g(y|x) \text{ and } \hat{f}(y|x) \in \Delta \cap \Xi\right)$$

$$= \int_X f_{\text{trg}}(x) \int_{Y|X} \hat{f}(y|x) \phi(x, y)$$

$$= \int_X f_{\text{trg}}(x) \int_{Y|X} \hat{f}(y|x) \log \frac{\hat{f}(y|x)}{f_o(y|x)}$$

$$= D(\hat{f}_\theta(Y|X)||f_o(Y|X))$$

The inequality holds because the Kullback-Leibler divergence is always non-negative with zero if and only if $g(y|x) = \hat{f}_\theta(y|x)$ almost everywhere, thus proving uniqueness.
Appendix A (Continued)

The non-negative constraint of equation 5.2.5 is superfluous since the log function in the objective function requires non-negative real numbers. Hence, the Lagrangian of the optimization problem is:

$$\mathcal{L}(\theta) = D_{f_{\text{trg}}(x), f(y|x)} \left( \hat{f}(Y|X) \parallel f_o(Y|X) \right) + \theta^T (E_{f_{\text{src}}(x)f(y|x)}[\Phi(X, Y)] - c)$$

$$= \int_X f_{\text{trg}} (x) \int_{Y|X} \hat{f}(y|x) \log \frac{\hat{f}(y|x)}{f_o(y|x)} + \theta^T (E_{f_{\text{src}}(x)f(y|x)}[\Phi(X, Y)] - c)$$

$$= \int_X f_{\text{trg}} (x) \int_{Y|X} \hat{f}(y|x) \left( \frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} \phi^T \Phi(x, y) \right) - \int_X f_{\text{trg}} (x) \int_{Y|X} \hat{f}(y|x) \log Z(x, \theta) +$$

$$\theta^T \left( \int_X f_{\text{src}}(x) \int_{Y|X} \hat{f}(y|x) \phi(x, y) - c \right)$$

$$= -\int_X f_{\text{trg}} (x) \log Z(x, \theta) - \theta^T c$$

Then:

$$\arg \max_\theta \mathcal{L}(\theta) = \arg \max_\theta \left( -\int_X f_{\text{trg}} (x) \log Z(x, \theta) - \theta^T c \right)$$

$$= \arg \max_\theta \left( -\int_X f_{\text{trg}} (x) \log Z(x, \theta) - \theta^T \int_X f_{\text{trg}} (x) \int_{Y|X} f(y|x) \phi(x, y) \right)$$

$$= \arg \max_\theta \left( -\int_X f_{\text{trg}} (x) \log Z(x, \theta) - \theta^T \int_X f_{\text{trg}} (x) \frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} \int_{Y|X} f(y|x) \phi(x, y) \right)$$

$$= \arg \max_\theta \left( \int_X f_{\text{trg}} (x) \int_{Y|X} f(y|x) \left( \log \frac{1}{Z(x, \theta)} - \frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} \phi^T \Phi(x, y) \right) \right)$$

$$= \arg \max_\theta E_{f_{\text{trg}}(x)f(y|x)} \left[ \log \frac{\hat{f}_\theta(Y|X)}{f_o(Y|X)} \right]$$

$$= \arg \max_\theta E_{f_{\text{trg}}(x)f(y|x)} \left[ \log \hat{f}_\theta(Y|X) \right]$$
Appendix A (Continued)

Proof of Theorem 5.3. Taking the partial derivative of $\mathcal{L}(\theta)$ with respect to $\theta$ (by Leibniz’s rule for differentiation under the integral sign):

$$
\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = -\int_X f_{\text{trg}}(x) \frac{1}{Z(x, \theta)} \frac{\partial Z(x, \theta)}{\partial \theta} - c
$$

$$
= -\int_X f_{\text{trg}}(x) \int_{\mathcal{Y}|\mathcal{X}} \hat{f}_\theta(-\frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} \Phi(x, y)) - c
$$

$$
= \mathbb{E}_{f_{\text{src}}(x)\hat{f}_\theta(y|x)} [\Phi(X, Y)] - c \quad (A.2.1)
$$

Proof of Corollary 5.1. Since

$$
f_o(y|x) \propto e^{-\frac{1}{2}y^T\Sigma_o^{-1}y + y^T\Sigma_o^{-1}\mu_o(x)},
$$

Then

$$
\hat{f}_\theta(y|x) \propto f_o(y|x) e^{-\frac{1}{2}y^T\Sigma_o^{-1}y + y^T\Sigma_o^{-1}\mu_o(x)}
$$

$$
-\frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)}\theta^T \begin{bmatrix} y \\ x \end{bmatrix} \begin{bmatrix} y \\ M \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix}
$$

$$
\propto e^{-\frac{1}{2}y^T(2\frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} M(y, y) + \Sigma_o^{-1})y + y^T(-2\frac{f_{\text{src}}(x)}{f_{\text{trg}}(x)} M(y, x_1) + \Sigma_o^{-1})\mu_o(x)} + 1
$$

Appendix A (Continued)

Proof of Theorem 5.4. When the new constraint is applied, the constrained optimization problem (Equation 5.2.5) becomes:

\[
\min_{f(Y|X)} D_{f_{\text{trg}}(x), f(y|x)} \left( \hat{f}(Y|X) || f_0(Y|X) \right)
\]

such that:
\[
E_{f_{\text{trg}}(x)f(y|x)}[\Phi(X, Y)] = \tilde{c}'
\]

Following the same procedure of solving the optimization problem in Theorem 5.2, the solution takes the form \( \hat{f}_\theta(y|x) = f_0(y|x)e^{-\theta^T \Phi(x, y)} \), with \( Z(x) = \int_{y \in Y} f_0(y|x)e^{-\theta^T \Phi(x, y)} \), which is not the same form as equation 5.2.6. Similar to the proof of Theorem 5.2, the Lagrangian after plugging in \( \hat{f}_\theta(y|x) \) becomes:

\[
\mathcal{L}(\theta) = D_{f_{\text{trg}}(x), f(y|x)} \left( \hat{f}(Y|X) || f_0(Y|X) \right) + \theta^T \left( \mathbb{E}_{f_{\text{trg}}(x)f(y|x)}[\Phi(X, Y)] - \tilde{c}' \right)
\]

\[
= - \mathbb{E}_{f_{\text{trg}}(x)}[\log Z(X, \theta)] - \theta^T \hat{c}'.
\]

Take the gradient with respect to \( \theta \), the gradient is

\[
\mathbb{E}_{f_{\text{trg}}(x)f(y|x)}[\Phi(X, Y)] - \tilde{c}' = \mathbb{E}_{f_{\text{src}}(x)f(y|x)} \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \Phi(X, Y) \right] - \tilde{c}',
\]

which becomes

\[
\mathbb{E}_{f_{\text{src}}(x)f(y|x)} \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \Phi(X, Y) \right] - \tilde{c}' - \lambda \theta,
\]
Appendix A (Continued)

when constraint slack and dual regularization is applied to allow for the noise from finite sample approximation.

We first prove the Lagrangian maximization problem above is equivalent to the reweighted conditional log likelihood maximization problem. The reweighted conditional log likelihood maximization is:

\[
ll(\theta) = \mathbb{E}_{f_{\text{src}}(x)} f(y|x) \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \log \frac{\hat{f}_{\theta}(y|X)}{f_0(y|X)} \right]
\]

\[
= -\mathbb{E}_{f_{\text{src}}(x)} \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \log Z(X, \theta) \right] - \theta^T \hat{c}'.
\]

If the same regularization is applied, the gradient with respect to \(\theta\) is

\[
\mathbb{E}_{f_{\text{src}}(x)} f(y|x) \left[ \frac{f_{\text{trg}}(X)}{f_{\text{src}}(X)} \Phi(X, Y) \right] - \hat{c}' - \lambda \theta,
\]

which is the same with the gradient of the Lagrangian maximization problem. Therefore, robust bias-aware regression is equivalent with reweighted conditional log likelihood maximization problem.

Furthermore, we will prove that when the feature function takes the quadratic form as in Corollary 5.1 reweighted conditional log likelihood maximization problem is equivalent with the importance weighted least square regression. Since \(f_0(y|x)\) is a Gaussian distribution \(N(\mu_0, \Sigma_0)\)
Appendix A (Continued)

and feature function takes quadratic form, the resulting distribution \( \hat{f}_M(y|x) \) is also a Gaussian, where

\[
\mu = (2M(y,y) + \sigma_0^{-2})^{-1}(-2M(y,x_1) + \sigma_0^{-2}\mu_0)
\]

\[
\sigma^2 = (2M(y,y) + \sigma_0^{-2})^{-1}
\]

\( \hat{f}_M(y|x) \) can take form \( N(a + b^T x, \sigma^2) \), the reweighted conditional log likelihood maximization problem is represented as:

\[
\mathbb{E}_{f_{src}(x)} f(y|x) \left[ \frac{1}{2} \frac{f_{trg}(X)}{f_{src}(X)} \log(2\pi\sigma^2) + \frac{f_{trg}(X)}{f_{src}(X)} \frac{(Y - a - b^T X)^2}{2\sigma^2} \right] + \mathbb{E}_{f_{src}(x)} f(y|x) \frac{f_{trg}(X)}{f_{src}(X)} \log(f_0(Y|X))
\]

If we consider \( \sigma^2 \) as a constant, minimizing the above function is equivalent with minimizing the reweighted squared loss \( \mathbb{E}_{f_{src}(x)} f(y|x) \frac{f_{trg}(X)}{f_{src}(X)} (Y - a - b^T X)^2 \).
Appendix B

COPYRIGHTS
Appendix B (Continued)

Proceedings of Machine Learning Research

ISSN: 1938-7228

The Proceedings of Machine Learning Research (formerly JMLR Workshop and Conference Proceedings) is a series aimed specifically at publishing machine learning research presented at workshops and conferences. Each volume is separately titled and associated with a particular workshop or conference and will be published online on the PMLR web site. Authors will retain copyright and individual volume editors are free to make additional hardcopy publishing arrangements (see for example the Challenges in Machine Learning series which includes free PDFs and low cost hard copies), but PMLR will not produce hardcopies of these volumes.

Editors

The Series Editors are Neil Lawrence and Mark Reid. Please send proposals for new volumes under this series to Neil and Mark via e-mail: mark_reid.name and N.Lawrence@sheffield.ac.uk. Each proposal should include:

- A brief description of the event’s scope and topics to be covered.
- A description of the review process for the proceedings.
- The names and short CVs (a few lines) of the proposed Proceedings Editors.

For frequently asked questions on preparing proceedings please see the FAQ.

Scheduled Volumes

Volume 77 is yet to be assigned
Volume 76 is assigned to the 2017 Algorithmic Learning Theory Conference (ALT 2017)
Volume 75 is assigned to the 2018 Conference on Learning Theory (COLT 2018)
Volume 74 is assigned to the ECML/PKDD 2017 Workshop on Learning with Imbalanced Domains: Theory and Applications (LIDTA 2017)
Volume 73 is assigned to the 2017 Conference on Advanced Methodologies in Bayesian Networks (AMBN 2017)
Volume 72 is assigned to the PGM 2018 Conference on Probabilistic Graphical Models
Volume 71 is assigned to the KDD 2017 Workshop on Anomaly Detection in Finance
Volume 68 is assigned to the 2017 Machine Learning in Health Care Conference (MLHC 2017)
Volume 70 is assigned to the 2017 International Conference on Machine Learning (ICML 2017)
Volume 69 is assigned to the KDD 2017 Workshop on Medical Informatics and Healthcare (MIH’17)
Volume 67 is assigned to the 3rd International Conference on Predictive Applications and APIs (PAPIs 2016)
Volume 66 is assigned to the IJCAI 2017 Workshop on Artificial Intelligence in Affective Computing
Volume 62 is assigned to International Symposium on Imprecise Probability: Theories and Applications (ISIPTA 2017)
Volume 61 is assigned to the NIPS 2016 Symposium on Machine Learning and the Law
Volume 59 is assigned to the NIPS 2016 Workshop on Reliable Machine Learning in the Wild
Volume 58 is assigned to the NIPS 2016 Workshop on Imperfect Decision Makers

Proceedings

Volume 70 Proceedings of ICML 2017
You hereby grant to the Journal of Machine Learning Research the following rights with respect to the article described below:

- exclusive right of first publication in all media
- right of subsequent publication in all media
- the right to sublicense or transfer this license to any publisher of Journal of Machine Learning Research.

This license will be valid throughout the world throughout the entire term of copyright in the article, and is granted free of royalty. You will ensure that any dissemination of the article authorized by you gives appropriate first publication credit to Journal of Machine Learning Research.

You explicitly reserve all other proprietary rights with respect to the article, including copyright and patent rights.

If there is more than one author of the article, the word “you” includes all authors jointly and severally. By signing this form, the signatory acknowledges that he or she is the corresponding author; that the signatory is signing on behalf of all authors of the article; and that the signatory has the authority to act as the agent for the other authors of the article.

You warrant that you are the sole author of the article, and generally that you have a complete and unencumbered right to make the grants you make to us. You also warrant that the article does not libel anyone, invade anyone’s copyright or otherwise violate any statutory or common law right of anyone, and that you have made all reasonable efforts to ensure the accuracy of any factual information contained in the article. You agree to indemnify us and any sublicensees against any claim or action alleging facts which, if true, constitute a breach of any of the foregoing warranties.

This is the entire agreement between us, and it may be modified only in writing. It will be governed by the laws of the Commonwealth of Massachusetts. It will bind and benefit our respective assigns and successors in interest, including your heirs. It will terminate if we do not publish, in any medium, your article within two years of the date of your signature(s).

Title of Article

Author name(s)

Name of Signatory

Signature

Date
Appendix B (Continued)

You hereby grant to the Journal of Machine Learning Research the following rights with respect to the article described below:

- exclusive right of first publication in all media
- right of subsequent publication in all media
- the right to sublicense or transfer this license to any publisher of Journal of Machine Learning Research.

This license will be valid throughout the world throughout the entire term of copyright in the article, and is granted free of royalty. You will ensure that any dissemination of the article authorized by you gives appropriate first publication credit to Journal of Machine Learning Research.

You explicitly reserve all other proprietary rights with respect to the article, including copyright and patent rights.

If there is more than one author of the article, the word "you" includes all authors jointly and severally. By signing this form, the signatory acknowledges that he or she is the corresponding author; that the signatory is signing on behalf of all authors of the article; and that the signatory has the authority to act as the agent for the other authors of the article.

You warrant that you are the sole author of the article, and generally that you have a complete and unencumbered right to make the grants you make to us. You also warrant that the article does not libel anyone, invade anyone's copyright or otherwise violate any statutory or common law right of anyone, and that you have made all reasonable efforts to ensure the accuracy of any factual information contained in the article. You agree to indemnify us and any sublicensees against any claim or action alleging facts which, if true, constitute a breach of any of the foregoing warranties.

This is the entire agreement between us, and it may be modified only in writing. It will be governed by the laws of the Commonwealth of Massachusetts. It will bind and benefit our respective assigns and successors in interest, including your heirs. It will terminate if we do not publish, in any medium, your article within two years of the date of your signature(s).

<table>
<thead>
<tr>
<th>Title of Article</th>
<th>Robust Covariate Shift Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author name(s)</td>
<td>Xiangli Chen; Mathew Monfort;</td>
</tr>
<tr>
<td></td>
<td>Anqi Liu; Brian D. Ziebart</td>
</tr>
<tr>
<td>Name of Signatory</td>
<td>Xiangli Chen</td>
</tr>
<tr>
<td>Signature</td>
<td>Xiangli Chen</td>
</tr>
<tr>
<td>Date</td>
<td>2016/04/17</td>
</tr>
</tbody>
</table>
Appendix B (Continued)

UAI/CoRR Copyright Permission

**TITLE OF WORK:** Adversarial Inverse Optimal Control for General Imitation Learning Losses and Environment Transfer

**AUTHOR(S):** Xiangli Chen; Mathew Monfort; Brian D. Ziebart; Peter Carr

**DESCRIPTION OF MATERIAL:** Paper in UAI 2016 proceedings

I hereby grant permission for AUAI and CoRR to include the above-named material (the Material) in the UAI and CoRR Digital Libraries.

[ ] Yes. I verify that I am the owner of copyright in this Material and have the authority to grant permission.

I hereby release and discharge AUAI, CoRR, and other publication sponsors and organizers from any and all liability arising out of my inclusion in the publication, or in connection with the performance of any of the activities described in this document as permitted herein. This includes, but is not limited to, my right of privacy or publicity, copyright, patent rights, trade secret rights, moral rights or trademark rights.

All permissions and releases granted by me herein shall be effective in perpetuity unless otherwise stipulated, and extend and apply to the AUAI, CoRR, and its assigns, contractors, sublicensees, distributors, successors and agents.

The following statement of copyright ownership will be displayed with the Material, unless otherwise specified: "Copyright is held by the author/owner."

In the event that any elements used in the Material contain the work of third-party individuals, I understand that it is my responsibility to secure any necessary permissions and/or licenses and will provide same in writing to AUAI and CoRR. If the copyright holder requires a citation to a copyrighted work, I have obtained the correct wording and have included it in the designated space in the text.

[ ] No. I have not used third-party material.

[ ] I have the necessary permission to use third-party material.

**SIGNATURE (author/owner)**

Xiangli Chen 05/23/2016

**PRINT NAME**

**DATE**

Note: Each contributor must submit a signed form, or indicate below that one agent signs for all co-authors.

[ ] I am signing on behalf of all co-authors

Rev. 3.11
CITED LITERATURE


174


VITA

NAME Xiangli Chen

EDUCATION

Ph.D. in Computer Science, University of Illinois at Chicago, 2017

Master in Computer Science, University of Illinois at Chicago, 2010

Bachelor in Software Engineering, XiDian University, 2008

PUBLICATIONS


Xiangli Chen, Mathew Monfort, Anqi Liu and Brian D.Ziebart. “Robust Covariate Shift Regression.” International Conference on Artificial Intelligence and Statistics (AISTATS), 2016

