universität wien

# BACHELORARBEIT / BACHELOR'S THESIS

Titel der Bachelorarbeit / Title of the Bachelor's Thesis

„Maximum Causal Entropy Inverse Reinforcement Learning: An Extension to Higher Order Moment Matching“


verfasst von / submitted by

Rebecca Weiß, BSc


angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Bachelor of Science (BSc)

Wien, 2024 / Vienna, 2024

| | |
|---|---|
| Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet: | UA 033 521 |
| Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet: | Bachelorstudium Informatik |
| Betreut von / Supervisor: | Assoz. Prof. Dipl.-Ing. Dr.techn. Sebastian Tschiatschek, BSc |

# Acknowledgements

I would like to extend my gratitude to those who have supported me throughout the completion of this thesis.

Special thanks to my thesis advisor, Professor Tschiatschek, for his guidance, encouragement, and insightful feedback throughout this research endeavor.

I am also thankful to my family and friends for their support and understanding during this academic journey.

Finally, I acknowledge the University of Vienna for providing the necessary resources and conducive environment for conducting this research.

# Abstract

This thesis extends Maximum Causal Entropy Inverse Reinforcement Learning (MCE IRL) approach by integrating constraints on higher order moments of the cumulative feature vectors alongside the original constraint of feature expectation matching, and can be generalized to higher-order moment matching. The extended algorithm aims to improve IRL to better reflect the demonstrator's behavior and maximize the learned policy's reward under the true unknown reward parameters.

In a grid world example, this thesis demonstrates superior performance over the original method by achieving closer alignment with the demonstrator's policy. Specifically, the algorithm closely matches the demonstrator's reward and cumulative feature variance under the true unknown reward parameters resulting in minimal error and better behavioral alignment than the original algorithm.

This thesis underlines the effectiveness of incorporating feature variance and higher-order moment matching in refining IRL algorithms for robust behavioral modeling in practical applications.

# Contents

# List of Figures

# List of Algorithms

# 1 Introduction

Inverse Reinforcement Learning (IRL) is a powerful and popular framework in machine learning, designed to uncover the unknown reward function that motivates an agent's behavior [1]. This framework is particularly useful in scenarios where only the demonstrations in the form of trajectories of a demonstrator's behavior are available, while the reward function is not explicitly known. A prominent method within IRL is the Maximum Causal Entropy IRL (MCE IRL)[2], which focuses on finding a policy that mimics the observed behavior well while simultaneously maximizing the entropy of the policy, i.e., the randomness of the policy. This prevents over-fitting to specific actions and often provides a more robust understanding of the demonstrator's decision making [3]. Furthermore, it can handle learning under imperfect behavior demonstrations [4]. This balance between accuracy and randomness makes MCE IRL a valuable tool for inferring reward structures from complex and potentially noisy data.

The underlying concept of MCE IRL is Feature Expectation Matching, which seeks to find a policy that matches the expected feature count of the demonstrator, approximated via the given behavior trajectories [2]. It can be beneficial to consider not only feature expectation values but also feature variances and potentially higher-order moments to make the process of simulating the demonstrator's behavior even more accurate.

The goal of this work is to extend the existing MCE IRL algorithm to match both the demonstrator's feature expectations and variances. This can be motivated by the fact, there might be multiple policies that match the demonstrator's feature expectation but all have different variances. Thus, further matching the feature variance can potentially lead to a better reflection of the demonstrator's behavior as can be shown in a simple grid-world experiment. We derive the Lagrangian of the new optimization problem which can be transformed into its dual problem. This dual problem will be computed by solving the nested optimization problem via soft value iteration. Aggregating all step, we define an algorithm that can be used to solve the original optimization problem with gradient descent.

The remainder of this thesis is structured as follows. In Chapter 2 and 3, we will first present some background information and notations, as well as related algorithms and approaches, to make this work more accessible for readers new to the field. We then present the theoretical foundation for the new algorithm, inspired by Gleave and Toyer's primer on MCE IRL [3] in Chapter 4, following the structure of their paper and extending the proofs to our new approach. With this theoretical basis, we will present the modified algorithm and provide a closed-form solution for the special setting of Linear Quadratic

Regulators (LQR).

We will discuss the challenges of extending the algorithm to infinite-horizon Markov Decision Processes and conclude the theoretical part of this work with a generalization to matching higher-order moments in Section 4.6, making this approach very flexible depending on the desired matching accuracy.

Finally, in Chapter 5, we present a small practical example to demonstrate the potential power of this work's contribution, offering insights into how to extend and apply this approach to more complex and intricate settings. We conclude this work and suggest interesting topics for future work in Chapter 6.

# 2 Background

In this section, we introduce relevant concepts to this work and our notation. We provide an overview of *Reinforcement Learning*, specifically *Inverse Reinforcement Learning*. We also discuss *Markov Decision Processes* and *Linear Quadratic Regulators*. Finally, we define *Causal Entropy* and its significance for this work.

## 2.1 Reinforcement Learning

Reinforcement Learning (RL) is a framework to learn from interaction. This type of learning is intuitive for humans. For example, as children interact with their environment, they learn the cause, and effect of actions and discover how to achieve goals based on feedback from their surroundings [1].

The main idea of the RL problem is for an agent to learn what to do in a certain scenario. Specifically, the agent learns how to map states to actions in order to maximize a numerical reward signal. To achieve this, the agent must interact with its environment, having a concept of the environment's current situation and a set of actions it can take. RL differs from *supervised learning* problems because the agent is not told what to do or how to proceed correctly; it must learn on its own which actions maximize its reward. RL is also distinct from *unsupervised learning* because there is no hidden structure in the data that the agent tries to find. Thus, Reinforcement Learning can be considered a third kind of learning paradigm [1].

Throughout its learning process, the agent learns a so-called policy, which stores information about how to proceed in certain states to succeed. Policies can be deterministic, i.e., the agent always takes the same action in a specific state, or stochastic, where its choice depends on the probability of all actions in a certain state. A key challenge in

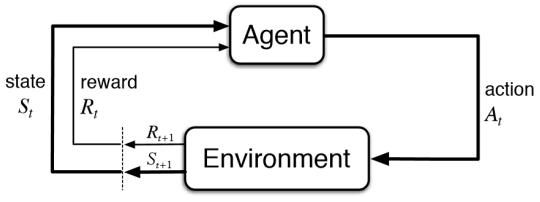


Figure 2.1: Typical RL setting [5]. The agent sends actions to the environment, the environment sends back a reward signal and information about the new state.

obtaining this policy is the trade-off between exploration and exploitation. To maximize its reward signal, the agent should take actions it has already learned to be effective. However, if it always does this, it cannot improve its current policy. Thus, exploration, where the agent takes either new strategic or random actions with a certain probability to observe new states and possibly find better strategies, should also be encouraged. It is crucial to balance exploration and exploitation [1].

## 2.2 Markov Decision Processes

Markov Decision Processes (MDPs) characterize RL environments that satisfy the Markov property, i.e., the environment's response at a certain time $t+1$ only depends on the state and action of the previous time step $t$. We use the formal definition from [3].

**Definition 1.** *A Markov Decision Process (MDP) $M = (\mathcal{S}, \mathcal{A}, \gamma, T, \mathcal{I}, \mathcal{T}, r)$ consists of a set of states $\mathcal{S}$ and a set of actions $\mathcal{A}$; a discount factor $\gamma \in [0,1]$; a horizon $T \in \mathbb{N} \cup \{\infty\}$; an initial state distribution $\mathcal{I}(s)$; a transition distribution $\mathcal{T}(s' \mid s, a)$ specifying the probability of transitioning to $s'$ from $s$ after taking action $a$; and a reward function $r(s, a, s')$ specifying the reward upon taking action $a$ in state $s$ and transitioning to state $s'$. It must be true that either the discount factor satisfies $\gamma < 1$ or that the horizon is finite ($T < \infty$).*

Note that MDPs can be both finite and discounted but must satisfy at least one of these properties. Given such an MDP, we can define a (stochastic) *policy* $\pi_t(a_t \mid s_t)$ that gives the probability of taking action $a_t \in \mathcal{A}$ in state $s_t \in \mathcal{S}$ at time step $t$. Within this setting, we can then observe behavior *trajectories* of a given demonstrator, i.e., sequences of alternating states and actions that represent the states the demonstrator has visited and action it has taken in each of the state at a certain time step. For a given *trajectory fragment* $\tau = (s_0, a_0, s_1, a_1, \ldots, s_{k-1}, a_{k-1}, s_k)$ of length $k \in \mathbb{N}$, the probability of a policy acting in the MDP producing $\tau$ is given by

$$p(\tau) = \mathcal{I}(s_o) \prod_{t=0}^{k-1} \mathcal{T}(s_{t+1} \mid s_t, a_t) \pi_t(a_t \mid s_t). \tag{2.1}$$

We use the same assumption as in [3], namely that in finite-horizon MDPs the policy can be *non-stationary* and thus can depend on the current time step $t$. In the case of infinite-horizon MDPs, due to their symmetry in time, the policy is stationary, and we can simply write $\pi$ and drop the dependency on $t$.

Finally, we define the goal of an agent, which is to maximize the expected total reward

$$G(\pi_t) = \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{t=0}^{T-1} \gamma^t r(S_t, A_t, S_{t+1}) \right] \tag{2.2}$$

Note, that $S_t$, $A_t$ and $S_{t+1}$ denote random variables that represent the states and actions at time step $t$ resp. $t+1$, whereas the possible actions at a specific time step $t$ are

given by $\pi_t$.

An optimal policy $\pi_t^*$ is then the policy that achieves the highest possible total reward: $\pi_t^* \in \arg\max_{\pi_t} G(\pi_t)$.

In order to find an optimal policy, we define the *state value function* $V$ which maps states to the expected (discounted) reward in the future starting from the given state. In addition to this function, we can also define the *action value function* $Q$ that maps pairs of states and actions to the expected (discounted) future reward starting from the given state and taking the given action [6].

Based on these definitions, we can find an optimal policy is by solving the Bellman equations

$$\begin{aligned} V^*(s_t) &= \max_{a_t}\{Q^*(s_t, a_t)\} \\ Q^*(s_t, a_t) &= r(s_t, a_t, s_{t+1}) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{T}(s' \mid s_t, a_t) V^*(s') \end{aligned} \tag{2.3}$$

where $V^*(s_t)$ is the optimal *state value function* and $Q^*(s_t, a_t)$ the optimal *action value function*. Equations 2.3 can be solved by iteratively updating $V^*$ with the so-called *value iteration* algorithm [4].

In this work, we will use the softened Bellman equations, which use a softmax instead of a max. More details can be found in Section 3.2.

## 2.3 Inverse Reinforcement Learning

While RL is effective for learning policies in environments with well-defined reward signals, it can be challenging or impossible to define a good reward function. A classic example is driving a vehicle, where numerous variables, often considered subconsciously by humans, influence actions, such as maintaining a safe distance from other cars, obeying speed limits, avoiding pedestrians, and personal driving style preferences. Specifying the impact of these variables numerically is difficult. Often, one would manually tweak the reward function until it produced the desired results [7].

To overcome this obstacle, apprenticeship learning was proposed. In apprenticeship learning, the agent observes the behavior of an "expert" and learns its policy from these trajectories, avoiding the need to manually tweak the reward signal [8].

Inverse Reinforcement Learning (IRL) is a type of apprenticeship learning where the agent tries to learn not just the policy but also the reward signal under which the expert operates. This is motivated by the presumption that the reward function is the "most succinct, robust, and transferable definition of the task" [8]. Once the reward signal is

found, the policy can be determined using any RL algorithm.

A significant challenge in IRL is that the problem is under-constrained, meaning there are many different reward functions consistent with the expert's behavior. Some differences do not change the optimal policy and can be ignored, such as scale and potential shaping [3]. However, some differences can indeed change the optimal policy, potentially only in states not covered by the expert demonstrations [3].

There are different approaches to addressing this ambiguity, which we will present in later sections, especially in Chapter 3.

## 2.4 Linear Quadratic Regulators

In this section, we briefly discuss Linear Quadratic Regulators (LQR). While we usually expect our reward function to be linear in the state-action feature representations, it can sometimes be useful to consider quadratic functions. In LQRs we consider such functions that have both linear and quadratic terms, but in this scenario we see them as cost functions and not reward function. We define LQRs as the following in [4].

**Definition 2.** *A Linear Quadratic Regulation setting is characterized by state dynamics that are distributed according to a linear function of past state and action variables with fixed parameters $A$ and $B$ and with Gaussian noise $\epsilon$. Cost functions are quadratic and linear in the state variables parameterized by $Q$ and $R$ respectively*

$$\begin{aligned} s_{t+1} &= As_t + Ba_t + \epsilon_t \\ \epsilon &\sim N(0, \Sigma) \\ Cost(s_{1:T}, a_{1:T}) &= \sum_{t=0}^{T} s_t^T Q s_t + s_t^T R \end{aligned} \tag{2.4}$$

*where the tuple $(s_{1:T}, a_{1:T})$ denotes the trajectory $(s_1, a_1, s_2, a_2, \dots)$.*

Note that the definition above speaks solely of cost function with respect to state variables as there is no explicit dependence on the actions $a_i$. Ziebart [4] mentions that such cost functions are possible for action variables but can always be represented as state-based costs, so the focus remains on state-based costs.

In Section 4.5, we will see that for such dynamics, our approach can be solved via soft value iteration and also allows for a closed-form solution, similar as in [4].

## 2.5 Maximum Causal Entropy

In the following chapter, we will explore approaches that use the principle of *maximum entropy* to address the under-constrained nature of the IRL problem. As the name implies,

this principle involves choosing the distribution with the highest entropy, and thus the highest uncertainty [3].

This principle can be used to minimize the worst-case prediction log-loss, providing a good guarantee for various machine learning techniques [4]. For example, in games, this principle is useful because it allows one to choose the distribution that minimizes the expected log loss in an environment where the opponent selects the true distribution from all those consistent with the data [3].

In our approach, we use the principle of *maximum causal entropy*, as employed by Ziebart [2, 3]. We will give the definition of causal entropy as it is provided in [3] below.

**Definition 3.** *Let $S_{0:T}$ and $A_{0:T}$ be random variables representing states and actions induced by following a policy $\pi_t$ in an MDP and sampled according to Eq. 2.1. Then the causal entropy $H(A_{0:T-1} \,||\, S_{0:T-1})$ is the sum of the entropies of the policy action selection conditioned on the state at that time step:*

$$H(A_{0:T-1} \,||\, S_{0:T-1}) = \mathop{\mathbb{E}}_{\pi_t}\left[ -\sum_{t=0}^{T-1} \gamma^t \log \pi_t(A_t \,|\, S_t) \right] = \sum_{t=0}^{T-1} \gamma^t H(A_t \,|\, S_t) \tag{2.5}$$

It can be shown that also the maximum causal entropy distribution minimizes the worst case prediction log-loss [2].

Causal entropy is motivated by conditioning only on the information available to the agent at each time step, i.e., its current state and previous states and actions. This differs from conditional entropy, which conditions on states that come up after each action was taken.

Shannon entropy, another example, calculates the entropy over the entire trajectory distribution, introducing a bias towards taking actions with uncertain outcomes that could potentially be risky [3]. In settings with stochastic processes where information is dynamic and only becomes available over time, this difference is crucial. In such settings, future states have no causal influence over earlier actions, making conditional maximum entropy ill-suited [3].

Therefore, causal entropy should be preferred, especially in stochastic MDPs, as Shannon entropy and causal entropy coincide for deterministic MDPs [3].

# 3 Related Work

In this chapter, we provide an overview of related approaches, focusing on *Maximum Causal Entropy IRL*, which is the basis for this work. There are several extensions of the basic concepts of Maximum Entropy and Maximum Causal Entropy IRL, e.g., [9, 10, 11, 12, 13, 14] but we will only focus on a few of them in this section.

## 3.1 Maximum Entropy IRL

Inverse Reinforcement Learning (IRL) aims to learn the reward function from a demonstrator's trajectories, ultimately leading to a policy that performs similarly to the demonstrations. This similarity is often formalized by matching *feature expectations*, specifically the expected feature counts. The motivation for this approach is that the reward function optimized by the demonstrator is assumed to be linear, i.e., of the form $r_{\theta_\star}(s_t, a_t) = \theta_\star^T \phi(s_t, a_t)$ with $\phi(s_t, a_t) \in \mathbb{R}^d$ being some features associated with states and actions. Thus

$$\mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t r_{\theta_\star}(S_t, A_t)\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \theta_\star^T \phi(S_t, A_t)\right] = \theta_\star^T \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right], \tag{3.1}$$

where $\mathcal{D}$ is the behavior distribution of the demonstrator, i.e., the probability distribution modeling the behavior of the demonstrator from which the trajectories are sampled.

If we have a policy $\pi_t$ that has the exact same expected feature counts as the demonstrator, i.e.,

$$\mathop{\mathbb{E}}_{\pi_t}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right], \tag{3.2}$$

the reward that is obtained by the policy $\pi_t$ under the true reward function must be equal to the reward obtained by the demonstrator. If we can show that our policy $\pi_t$ is optimal under some reward function parameters $\hat{\theta}$, it is reasonable to say that $\hat{\theta}$ approximates the true unknown reward function parameters $\theta_\star$, at least up to some scaling and shifting constants as those do not change the optimal policy [15]. However, these parameters $\hat{\theta}$ are not unique due to the under-constrained nature of the IRL problem, necessitating a solution to this problem [3].

Maximum Entropy IRL resolves ambiguities in choosing a distribution over decisions by maximizing the entropy, as implied by the name. It further provides a method to deal with imperfections in the demonstrations of the "expert", i.e., the approach can deal with

trajectories that are sub-optimal due to noise and imperfect behavior of the expert [16].

The motivation behind the principle of maximum entropy is to choose the distribution that does not exhibit any additional preferences beyond matching the feature expectations [16].

## 3.2 Maximum Causal Entropy (MCE) IRL

Maximum Causal Entropy is very similar to the Maximum Entropy approach with the difference of using causal entropy instead of the standard entropy. Therefore also in this approach feature expectations/counts are used to measure similarity between the found policy and that of the demonstrator.

The use of causal entropy is motivated by the existence of environments in which side information is dynamic, i.e., it is revealed sequentially over time. In order to model this, causally conditioned probabilities can be used. Conditional maximum entropy, on the other hand, assumes all side information to be available upfront, making it unsuitable for environments where future states do not causally influence previous actions [2].

As mentioned in Section 2.2, one can use value iteration to find an optimal policy for an MDP. Using the maximum causal entropy approach, the softened Bellman equations are used instead:

$$\begin{aligned} V_{\theta,t}^{\text{soft}}(s_t) &= \log \sum_{a_t \in \mathcal{A}} \exp Q_{\theta,t}^{\text{soft}}(s_t, a_t) && (\forall\, 0 \le t \le T-1) \\ Q_{\theta,t}^{\text{soft}}(s_t, a_t) &= \theta^T \phi(s_t, a_t) + \gamma \mathop{\mathbb{E}}_{\mathcal{T}}\left[ V_{\theta,t+1}^{\text{soft}}(S_{t+1}) \,\middle|\, s_t, a_t \right] && (\forall\, 0 \le t < T-1) \\ Q_{\theta,T-1}^{\text{soft}}(s_{T-1}, a_{T-1}) &= \theta^T \phi(s_{T-1}, a_{T-1}) \end{aligned} \tag{3.3}$$

In this work, we will adapt these equations to our setting.

As a closing mentioning of this section it is to be observed that in the setting of LQRs, as presented in Section 2.4, one can find a closed-form solution for the Maximum Causal Entropy problem based on the soft value iteration in 3.3, as shown in [4, Theorem 6.15]. We will also demonstrate this property for the extension to feature variance matching.

## 3.3 Weighted Maximum Entropy IRL

An adaptation of the MCE IRL approach is Weighted Maximum Entropy IRL [11]. This approach uses a weight function to learn the demonstrator's policy's stochasticity and the structure of the entropy terms added to the MDPs. The authors define the adapted

weighted causal entropy problem as:

$$\max_{\pi_t} - \mathop{\mathbb{E}}_{\pi_t}\left[\sum_{t=0}^{T-1} \gamma^t \mu(s_t) \log \pi_t(a_t \,|\, s_t)\right] \tag{3.4}$$

$$\text{subject to } \mathop{\mathbb{E}}_{\pi_t}\left[\sum_{t=0}^{T-1} \gamma^t r(a_t \,|\, s_t)\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t r(a_t \,|\, s_t)\right], \tag{3.5}$$

where the $\mu(s_t)$ represent the weight function that is associated with each of the entropy terms $\log \pi_t(a_t \,|\, s_t)$.

This addresses the drawback of regular MCE IRL, where optimal policies tend to have similar stochasticity levels across all states, which may oversimplify real-life scenarios where experts' decisions can be more random in certain states than in others [11].

Weighted Maximum Entropy IRL learns the reward function, policy, and a function describing the policy's stochasticity. Intuitively, this function's value should be small when the policy in a state is more deterministic and larger when there is more randomness in the policy. Bui et. al. [11] provide constrained MDP formulations and keep the framework simple and general such that it can be incorporated into any IRL algorithm that is based on maximum entropy and therefore potentially also into the approach of this work.

In their formulation, weights $\mu(s_t)$ are added to the state-dependent entropies $H(A_t \,|\, S_t)$. The Kullback-Leibler divergence measures the closeness between a policy at a certain state and a random-walk policy. In the MDP formulation, the expected value of this divergence, is multiplied by the state-dependent weights $\mu(s_t)$ and bounded from above by a constant $\alpha$. Thus, if the weight is large, the divergence should be small, making the policy more random. For smaller weights, the divergence can be larger, making the policy more deterministic. This formulation generalizes the Maximum Causal Entropy approach, as it no longer fixes the entropy term structure but allows it to depend on the weights $\mu(s_t)$ [11].

## 3.4 Maximum Entropy Deep Inverse Reinforcement Learning

Maximum Entropy Deep IRL uses neural networks to learn complex, non-linear reward functions. As this method's training time does not depend on the number of given demonstration trajectories, it is advantageous to be scaled to problems with large state spaces as a lot of observations are needed in such settings in order to learn effectively. Additionally, using neural networks allows for an end-to-end learning scenario from raw input data to a reward function without needing complex feature representation design [12].

Neural networks (NNs) are particularly useful for learning arbitrary functions, as even shallow networks with only two layers and sigmoid activation functions can represent any binary or piecewise-linear function given enough nodes per layer. While this

result is powerful, deep architectures with many layers are usually preferred, as they reduce the number of necessary computations while often still providing similar results [12].

Wulfmeier et al. [12] use fully convolutional neural networks (FCNN) to preserve spatial information and create an output of the same spatial dimension and size as the input. Using this approach, one can see that the original MCE IRL method [2] can be implemented using a neural network with a single linear output connected to all input nodes with zero bias, solving the problem using classic backpropagation. The authors also include a term in in their loss function to encourage maximizing entropy to distinguish between optimal policies that obtain the same expected feature counts.

Thus, Deep NNs are particularly helpful for learning complex reward functions while maintaining a keeping the computational complexity relatively low w.r.t. the number of demonstration trajectories.

# 4 Feature Expectation and Variance Matching in MCE IRL

In this chapter, we present the main theoretical results of this work. We extend the approach of Maximum Causal Entropy (MCE) IRL by Ziebart et al. [2] by considering not only expected feature values but also their variance. We will explore how this change affects the derivations and proofs for MCE IRL and introduce an adapted algorithm that incorporates variance matching. Additionally, we will demonstrate that a closed-form solution for Linear Quadratic Regulators (LQRs) can still be found, making this approach at least as powerful as the original method. Lastly, we will sketch the generalization to higher-order moments.

We follow a structure similar to that of [3], extending all proofs to our setting to ensure clarity for readers new to the field.

## 4.1 Imitation as Feature Expectation and Variance Matching

In [3], the motivation behind using feature expectation matching was to formally define what it means for our learned policy to resemble the demonstrator's behavior. However, it can be argued that considering feature variance might provide an even more accurate representation, as multiple policies with the same expected feature counts can have different variances. We present an example in Chapter 5 illustrating the significance of not only matching the feature expectations.

Assuming the demonstrator's unknown reward function is linear in the feature representation, i.e., of the form $r_{\theta_\star}(s_t, a_t) = \theta_\star^T \phi(s_t, a_t)$ with $\theta_\star \in \mathbb{R}^d$, its expected return under its behavior distribution $\mathcal{D}$ is linear in the expected sum of discounted feature vectors. For variance, we can do something similar, but it leads to a quadratic term.

$$\mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t r_{\theta_\star}(S_t, A_t)\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \theta_\star^T \phi(S_t, A_t)\right] = \theta_\star^T \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right] \tag{4.1}$$

$$\mathop{\mathbb{V}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t r_{\theta_\star} \phi(S_t, A_t)\right] = \mathop{\mathbb{V}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \theta_\star^T \phi(S_t, A_t)\right] = \theta_\star^T \mathop{\mathbb{V}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right] \theta_\star \tag{4.2}$$

Thus, similar to the argument in [3], we aim to find a policy $\pi_t$ such that

$$\mathop{\mathbb{E}}_{\pi_t}\left[\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)\right] \quad \text{and} \tag{4.3}$$

$$\mathop{\mathbb{V}}_{\pi_t}\left[\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)\right] = \mathop{\mathbb{V}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)\right] \tag{4.4}$$

where $\mathbb{E}_{\pi_t}$ is the expectation value given the probability distribution of the policy $\pi_t$ resp. $\mathbb{E}_{\mathcal{D}}$ the expectation value of the probability distribution of the demonstrator $\mathcal{D}$. The same holds for the variance terms.

Due to the linearity of the expected reward in the (matched) feature expectations, the reward obtained by a policy $\pi_t$ satisfying Eq. 4.3 and 4.4 must be the same as that of the demonstrator. Thus, the parameters $\hat{\theta}$, for which such a policy $\pi_t$ satisfying the given constraints is optimal, can be seen as an approximation for the true parameters $\theta_\star$, at least up to some scaling and shifting constants as they do not influence the optimality of the policy [3].

Further, it can be observed in a setting using quadratic reward/cost functions of the form $r_{\Theta_\star}(s_t,a_t) = \phi(s_t,a_t)^T\Theta_\star\phi(s_t,a_t)$ with $\Theta_\star \in \mathbb{R}^{d\times d}$ matching the feature variance leads to matching the quadratic moments

$$\mathbb{E}\Big[\Big(\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)\Big)\Big(\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)\Big)^T\Big] = \sum_{i,j=0}^{T-1}\gamma^{i+j}\mathbb{E}\Big[\phi(S_i,A_i)\phi(S_j,A_j)^T\Big] \tag{4.5}$$

due to the property $\mathbb{V}[X] = \mathbb{E}[XX^T] - \mathbb{E}[X]\mathbb{E}[X]^T$ and with this, we especially match every $\mathbb{E}\Big[\phi(S_i,A_i)_k\phi(S_j,A_j)_l\Big]$ $\forall i,j \in \{0,\dots,T-1\}$, $\forall k,l \in \{1,\dots,d\}$. On the other hand, we observe

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1}\gamma^t r_{\theta_\star}(S_t,A_t)\right] &= \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1}\gamma^t\phi(S_t,A_t)^T\Theta_\star\phi(S_t,A_t)\right] \\ &= \sum_{t=0}^{T-1}\gamma^t\sum_{i,j=1}^{d}(\Theta_\star)_{ij}\mathop{\mathbb{E}}_{\mathcal{D}}\Big[\phi(S_t,A_t)_i\phi(S_t,A_t)_j\Big]\end{aligned} \tag{4.6}$$

Due to the variance matching it follows that for any policy $\pi_t$ that satisfies the constraints in Eq. 4.3 and 4.4, it holds that for all $t \in \{0,\dots,T-1\}$

$$\mathop{\mathbb{E}}_{\pi_t}\Big[\phi(S_t,A_t)_i\phi(S_t,A_t)_j\Big] = \mathop{\mathbb{E}}_{\mathcal{D}}\Big[\phi(S_t,A_t)_i\phi(S_t,A_t)_j\Big] \quad \forall i,j \in \{1,\dots,d\} \tag{4.7}$$

Therefore matching the feature variance ensures an equivalent performance to the demonstrator under quadratic reward/cost functions.

We can now formulate the problem that we want to solve within this work, namely maximizing causal entropy under the constraints of feature expectation and variance matching.

While the original approach was introduced for a general setting of non-Markovian dynamics and policies, we will focus on Markovian dynamics and policies as well as feature functions that decompose across state-action transitions. Similar to [3], we will further assume that the horizon $T$ is finite, and we consider time-dependent (non-stationary) policies.

First, we state our optimization problem which is similar to that defined in [3] with added feature variance matching constraint.

**Optimization Problem 1** (MCE IRL primal problem)**.** *Given an expert data distribution $\mathcal{D}$, the optimization problem of finding a time-dependent stochastic policy $\pi_t(s_t, a_t)$ that matches the feature expectations and variances while maximizing causal entropy is*

$$\max_{\pi_t \in \mathscr{P}} \quad H(A_{0:T-1} \,||\, S_{0:T-1}) \tag{4.8}$$

$$\textit{subject to} \quad \mathop{\mathbb{E}}_{\pi_t}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right] \tag{4.9}$$

$$\begin{aligned} \textit{and} \quad & \mathop{\mathbb{E}}_{\pi_t}\left[\left(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right)\left(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right)^T\right] \\ & = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\left(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right)\left(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\right)^T\right] \end{aligned} \tag{4.10}$$

*where $\mathscr{P}$ denotes the set of all time-dependent stochastic policies, i.e.,*

$$\pi_t \in \mathscr{P} \iff \pi_t(a_t \,|\, s_t) \geq 0 \;\; \textit{and} \;\; \sum_{a'} \pi_t(a' \,|\, s_t) = 1 \quad (\forall 0 \leq t < T,\, \forall a_t \in \mathcal{A},\, \forall s_t \in \mathcal{S}), \tag{4.11}$$

*and $S_{0:T-1}$ and $A_{0:T-1}$ are random sequences of states and actions, induced by $\pi_t$ and sampled according to Eq. 2.1 on the left, and $\mathcal{D}$ on the right.*

Note that in Eq. 4.10, we made use of the property $\mathbb{V}(X) = \mathbb{E}(XX^T) - \mathbb{E}(X)\mathbb{E}(X)^T$. As we are already matching feature expectations $\mathbb{E}(X)$ in Eq. 4.9 the only new part within the variance that we have to match is $\mathbb{E}(XX^T)$, which is why only this term is appearing in the problem definition above.

## 4.2 Solving the Optimization Problem

This section will derive the algorithm that can be used in order to solve Optimization Problem 1. We use a standard approach for solving convex optimization problems, even though it is not convex in general, but the approach usually still works well in practice. Namely, we will use the following three steps as motivated in [3]:

1. **Form the Lagrangian**: We will transform our feature expectation and variance matching terms into weighted penalty terms.

2. **Derive the dual problem**: We will form the dual problem based on the Lagrangian that will introduce additional parameters, where those corresponding to the expectation matching can be interpreted as weights for the reward function.

3. **Dual ascent**: We will solve the dual problem using *dual ascent* alternates between maximizing the Lagrangian w.r.t. to the policy and minimizing the Lagrangian w.r.t. the dual variables until convergence.

### 4.2.1 The Lagrangian and the Dual Problem

Define the Lagrangian $\Lambda : \mathscr{P} \times \mathbb{R}^d \times \mathbb{R}^{d \times d} \to \mathbb{R}$ for Optimization Problem 1:

$$\begin{aligned}
\Lambda(\pi_t, \theta_E, \theta_V) &= H(A_{0:T-1} \,||\, S_{0:T-1}) && (4.12)\\
&+ \Big\langle \theta_E \,,\, \mathop{\mathbb{E}}_{\pi_t}\Big[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\Big] - \mathop{\mathbb{E}}_{\mathcal{D}}\Big[\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\Big]\Big\rangle && (4.13)\\
&+ \Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{\pi_t}\Big[\Big(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\Big)\Big(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\Big)^T\Big] \\
&- \mathop{\mathbb{E}}_{\mathcal{D}}\Big[\Big(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\Big)\Big(\sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t)\Big)^T\Big]\Big\rangle && (4.14)
\end{aligned}$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product, which coincides with the usual dot-product for vectors/column matrices. We assume further that $\theta_V$ is symmetric. This assumption is reasonable due to the symmetric nature of the variance and especially of the quadratic term that we use in our problem definition. Note that the constraint $\pi_t \in \mathscr{P}$ is an important one that we will have to pay attention to when forming a nested optimization problem in order to compute $\pi_t$ in a later step.

With this definition of the Lagrangian, we can now define the dual problem, which is very similar to the one in [3], only adapting the additional parameters for the variance matching penalty term.

**Optimization Problem 2** (Dual MCE IRL problem)**.** *Define the dual function* $g(\theta_E, \theta_V)$ *as*

$$g(\theta_E, \theta_V) = \max_{\pi_t \in \mathscr{P}} \Lambda(\pi_t, \theta_E, \theta_V). \tag{4.15}$$

*The dual MCE IRL problem is the problem of finding* $\pi_t^*$*,* $\theta_E^*$ *and* $\theta_V^*$ *such that* $\theta_E^*, \theta_V^*$ *attains*

$$g(\theta_E^*, \theta_V^*) = \min_{\substack{\theta_E \in \mathbb{R}^d \\ \theta_V \in \mathbb{R}^{d \times d}}} g(\theta_E, \theta_V), \tag{4.16}$$

*while* $\pi_t^*$ *attains*

$$\Lambda(\pi_t^*, \theta_E^*, \theta_V^*) = g(\theta_E^*, \theta_V^*). \tag{4.17}$$

We can solve this problem instead of its primal and then use the recovered policy $\pi_t^*$ as a solution for the primal Optimization Problem 1.

### 4.2.2 The Dual Function

As the Optimization Problem 2 is formulated using the dual function $g(\theta_E, \theta_V) = \max_{\pi_t} \Lambda(\pi_t, \theta_E, \theta_V)$, we can compute its value for given $\theta_E, \theta_V$ by solving a nested optimization problem over $\pi_t$ under the constraint that $\pi_t \in \mathscr{P}$. As already defined in Eq. 4.11, the simplex defining constraints for $\mathscr{P}$ are:

$$\pi_t(a_t \,|\, s_t) \geq 0 \;\text{ and }\; \sum_{a'} \pi_t(a' \,|\, s_t) = 1 \quad (\forall 0 \leq t < T,\, \forall a_t \in \mathcal{A},\, \forall s_t \in \mathcal{S}) \tag{4.18}$$

Causal entropy is not defined for negative elements in $\pi_t$ as we would then apply the logarithm to it, so the non-negativity constraints can be implicitly assumed, and we only focus on the normalization constraints. We will write these constraints as $h_{s_t}(\pi_t) = 0$ whereas

$$h_{s_t}(\pi_t) = \sum_{a \in \mathcal{A}} \pi_t(a \,|\, s_t) - 1 \tag{4.19}$$

Based on Eq. 4.15 and the constraints in Eq. 4.19 which ensure $\pi_t \in \mathscr{P}$, we can formulate the nested optimization problem for computing $g(\theta_E, \theta_V)$ for given $\theta_E$, $\theta_V$ as the following

**Optimization Problem 3** (Dual function primal problem)**.** *The problem of computing the dual function can be equivalently expressed as*

$$\max_{\pi_t \in \mathscr{P}} \quad \Lambda(\pi_t, \theta_E, \theta_V) \tag{4.20}$$

$$\textit{subject to} \quad h_{s_t}(\pi_t) = 0 \quad (\forall 0 \leq t < T,\, \forall s_t \in \mathcal{S}) \tag{4.21}$$

The Lagrangian for Optimization Problem 3 can be defined as

$$\Psi(\pi_t, \mu; \theta_E, \theta_V) = \Lambda(\pi_t, \theta_E, \theta_V) + \sum_{s_t \in \mathcal{S}, 0 \leq t < T} \mu_{s_t} h_{s_t}(\pi_t) \tag{4.22}$$

$$= H(A_{0:T-1} \,||\, S_{0:T-1}) \tag{4.23}$$

$$+ \Big\langle \theta_E \,,\, \mathop{\mathbb{E}}_{\pi_t} \Big[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big] - \mathop{\mathbb{E}}_{\mathcal{D}} \Big[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big] \Big\rangle \tag{4.24}$$

$$+ \Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{\pi_t} \Big[ \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big) \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big)^T \Big]$$

$$- \mathop{\mathbb{E}}_{\mathcal{D}} \Big[ \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big) \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big)^T \Big] \Big\rangle \tag{4.25}$$

$$+ \sum_{s_t \in \mathcal{S}, 0 \leq t < T} \mu_{s_t} \Big( \sum_{a \in \mathcal{A}} \pi_t(a \,|\, s_t) - 1 \Big) \tag{4.26}$$

where we introduce $\{\mu_{s_t}\}_{s_t \in \mathcal{S}, 0 \leq t < T}$ as dual variables for the normalization constraints.

To solve this problem, we employ the standard Karush-Kuhn-Tucker (KKT) conditions:

$$\nabla_{\pi_t} \Psi(\pi_t, \mu; \theta_E, \theta_V) = 0 \tag{4.27}$$

$$\nabla_{\mu_{s_t}} \Psi(\pi_t, \mu; \theta_E, \theta_V) = h_{s_t}(\pi_t) = 0 \quad (\forall 0 \leq t < T,\ \forall s_t \in \mathcal{S}) \tag{4.28}$$

We can observe that $\mu$ only appears in the first equation. We can set it to a value that ensures the satisfaction of the normalization constraints and the vanishing of the gradient [3]. We will see this later in our derivations. Next, we calculate the gradient of the Lagrangian for Optimization Problem 3 to solve the KKT conditions.

**Lemma 1.** *The gradient of $\Psi(\pi_t, \mu; \theta_E, \theta_V)$ is given by*

$$\begin{aligned}
&\nabla_{\pi_t(a_t \,|\, s_t)} \Psi(\pi_t, \mu; \theta_E, \theta_V) = \\
&\quad \mu_{s_t} - \rho_{\pi_t, t}(s_t) \Bigg( 1 + \log \pi_t(a_t \,|\, s_t) - \langle \theta_E \,,\, \phi(s_t, a_t) \rangle - \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\
&\quad - \mathop{\mathbb{E}}_{\pi_t} \Bigg[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \Big( \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle + \gamma^{t'} \langle \theta_V \cdot \phi(S_{t'}, A_{t'}) \,,\, \phi(S_{t'}, A_{t'}) \rangle - \log \pi_{t'}(A_{t'} \,|\, S_{t'}) \Big) \,\Bigg|\, s_t, a_t \Bigg] \\
&\quad - 2 \cdot \mathop{\mathbb{E}}_{\pi_t} \Bigg[ \sum_{i=t}^{T-1} \sum_{j=0}^{i-1} \gamma^{i+j-t} \langle \theta_V \cdot \phi(S_i, A_i) \,,\, \phi(S_j, A_j) \rangle \,\Bigg|\, s_t, a_t \Bigg] \Bigg)
\end{aligned} \tag{4.29}$$

*where*

$$\rho_{\pi_t, t}(s_t) = \gamma^t \mathbb{E}_{\pi_t}[\mathbb{I}[S_t = s_t]] = \gamma^t \mathbb{P}_{\pi_t}(S_t = s_t) \tag{4.30}$$

*is the discounted probability that the agent will be in state $s_t$ at time $t$ if it follows policy $\pi_t$.*

*Proof.* We derive the gradient, similar to [3] term-by-term. The detailed derivations for most of the terms can be found in [3] and therefore we won't cover them in this paper but only state the results. Thus, we get for the first few terms:

$$\nabla_{\pi_t(a_t \,|\, s_t)} \sum_{t'=0}^{T-1} \sum_{s_{t'} \in \mathcal{S}} \mu_{s_{t'}} \left( \sum_{a' \in \mathcal{A}} \pi_t(a' \,|\, s_t) - 1 \right) = \mu_{s_t} \tag{4.31}$$

$$\begin{aligned}
&\nabla_{\pi_t(a_t \,|\, s_t)} \Big\langle \theta_E \,,\, \mathop{\mathbb{E}}_{\pi_t} \Big[ \sum_{t'=0}^{T-1} \gamma^{t'} \phi(S_{t'}, A_{t'}) \Big] - \mathop{\mathbb{E}}_{\mathcal{D}} \Big[ \sum_{t'=0}^{T-1} \gamma^{t'} \phi(S_{t'}, A_{t'}) \Big] \Big\rangle \\
&= \rho_{\pi_t, t}(s_t) \Big\langle \theta_E \,,\, \phi(s_t, a_t) + \mathop{\mathbb{E}}_{\pi_t} \Big[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \phi(S_{t'}, A_{t'}) \,\Big|\, s_t, a_t \Big] \Big\rangle \\
&= \rho_{\pi_t, t}(s_t) \Bigg( \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \mathop{\mathbb{E}}_{\pi_t} \Big[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle \,\Big|\, s_t, a_t \Big] \Bigg)
\end{aligned} \tag{4.32}$$

$$\begin{aligned}
&\nabla_{\pi_t(a_t \,|\, s_t)} H(A_{0:T-1} \,||\, S_{0:T-1}) = \\
&- \rho_{\pi_t, t}(s_t) \Big[ 1 + \log \pi_t(a_t \,|\, s_t) + \mathop{\mathbb{E}}_{\pi_t} \Big[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \log \pi_{t'}(A_{t'} \,|\, S_{t'}) \,\Big|\, s_t, a_t \Big] \Big]
\end{aligned} \tag{4.33}$$

The interesting new part here is the derivation of the derivative of the feature variance matching term:

$$
\begin{aligned}
\nabla_{\pi_t(a_t\,|\,s_t)}\Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{\pi_t}\Big[ \big(\sum_{t'=0}^{T-1} \gamma^{t'}\phi(S_{t'},A_{t'})\big)\big(\sum_{t'=0}^{T-1} \gamma^{t'}\phi(S_{t'},A_{t'})\big)^T \Big] \\
- \mathop{\mathbb{E}}_{\mathcal{D}}\Big[ \big(\sum_{t=0}^{T-1} \gamma^{t}\phi(S_{t},A_{t})\big)\big(\sum_{t=0}^{T-1} \gamma^{t}\phi(S_{t},A_{t})\big)^T \Big]\Big\rangle \\
= \nabla_{\pi_t(a_t\,|\,s_t)}\Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{i,j=0}^{T-1} \gamma^{i+j}\phi(S_i,A_i)\phi(S_j,A_j)^T \Big]\Big\rangle && (4.34) \\
= \Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{S_t\sim\pi_t}\Big[ \nabla_{\pi_t(a_t\,|\,s_t)} \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j} \underbrace{\phi(S_i,A_i)\phi(S_j,A_j)^T}_{=:\Phi_{ij}} \,\Big|\, S_t \Big]\Big]\Big\rangle && (4.35) \\
= \Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{S_t\sim\pi_t}\Big[ \gamma^t \mathbb{I}[S_t = s_t] \nabla_{\pi_t(a_t\,|\,s_t)} \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j-t}\Phi_{ij} \,\Big|\, S_t = s_t \Big]\Big]\Big\rangle && (4.36) \\
= \rho_{\pi_t,t}(s_t)\Big\langle \theta_V \,,\, \nabla_{\pi_t(a_t\,|\,s_t)} \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j-t}\Phi_{ij} \,\Big|\, S_t = s_t \Big]\Big\rangle && (4.37) \\
= \rho_{\pi_t,t}(s_t)\Big\langle \theta_V \,,\, \nabla_{\pi_t(a_t\,|\,s_t)}\Big( \pi_t(a_t\,|\,s_t) \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j-t}\Phi_{ij} \,\Big|\, S_t = s_t, A_t = a_t \Big] \\
+ \Big( \sum_{a'_t\neq a_t} \pi_t(a'_t\,|\,s_t)\Big) \cdot \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j-t}\Phi_{ij} \,\Big|\, S_t = s_t, A_t \neq a_t \Big]\Big)\Big\rangle && (4.38) \\
= \rho_{\pi_t,t}(s_t)\Big\langle \theta_V \,,\, \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j-t}\Phi_{ij} \,\Big|\, s_t, a_t \Big]\Big\rangle && (4.39) \\
= \rho_{\pi_t,t}(s_t) \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{\substack{i,j=0 \\ i\geq t \vee j\geq t}}^{T-1} \gamma^{i+j-t}\langle \theta_V \cdot \phi(S_i,A_i)\,,\,\phi(S_j,A_j)\rangle \,\Big|\, s_t, a_t \Big] && (4.40) \\
= \rho_{\pi_t,t}(s_t)\Big( \gamma^t \langle \theta_V \cdot \phi(s_t,a_t)\,,\,\phi(s_t,a_t)\rangle \\
+ \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{t'=t+1}^{T-1} \gamma^{2t'-t}\langle \theta_V \cdot \phi(S_{t'},A_{t'})\,,\,\phi(S_{t'},A_{t'})\rangle \,\Big|\, s_t, a_t \Big] \\
+ 2\cdot \mathop{\mathbb{E}}_{\pi_t}\Big[ \sum_{i=t}^{T-1}\sum_{j=0}^{i-1} \gamma^{i+j-t}\langle \theta_V \cdot \phi(S_i,A_i)\,,\,\phi(S_j,A_j)\rangle \,\Big|\, s_t, a_t \Big]\Big) && (4.41)
\end{aligned}
$$

Where the last step is just splitting up the double sum into the given components and using properties of the Frobenius inner product, together with the assumption that $\theta_V$ is symmetric.

Collecting all terms together, we finally get the derivative of $\Psi(\pi_t, \mu; \theta_E, \theta_V)$ w.r.t. to the given policy as

$$\nabla_{\pi_t(a_t \mid s_t)} \Psi(\pi_t, \mu; \theta_E, \theta_V) = \tag{4.42}$$

$$\begin{aligned}
&\mu_{s_t} - \rho_{\pi_t,t}(s_t) \Bigg( 1 + \log \pi_t(a_t \mid s_t) - \langle \theta_E \,,\, \phi(s_t, a_t) \rangle - \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\
&- \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \Big( \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle + \gamma^{t'} \langle \theta_V \cdot \phi(S_{t'}, A_{t'}) \,,\, \phi(S_{t'}, A_{t'}) \rangle - \log \pi_{t'}(A_{t'} \mid S_{t'}) \Big) \,\middle|\, s_t, a_t \right] \\
&- 2 \cdot \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{i=t}^{T-1} \sum_{j=0}^{i-1} \gamma^{i+j-t} \langle \theta_V \cdot \phi(S_i, A_i) \,,\, \phi(S_j, A_j) \rangle \,\middle|\, s_t, a_t \right] \Bigg)
\end{aligned} \tag{4.43}$$

□

In order to solve the KKT condition that was mentioned above in Eq. 4.27, we can set the gradient that we found in Lemma 1 to zero and find a solution for $\pi_t$ that will satisfy this condition. This policy $\pi_t$ will be the one that gives us $g(\theta_E, \theta_V) = \Lambda(\pi_t, \theta_E, \theta_V)$. This policy can be computed using *soft value iteration*, as Lemma 2 will show.

**Lemma 2.** *The KKT condition $\nabla_{\pi_t} \Psi(\pi_t, \mu; \theta_E, \theta_V)$ is satisfied by a policy*

$$\pi_t(a_t \mid s_t) = \exp\left( Q^{soft}_{\theta_E,\theta_V,t}(s_t, a_t) - V^{soft}_{\theta_E,\theta_V,t}(s_t) \right) \tag{4.44}$$

*satisfying the following recursion:*

$$\begin{aligned}
V^{soft}_{\theta_E,\theta_V,t}(s_t) &= \log \sum_{a_t \in \mathcal{A}} \exp Q^{soft}_{\theta_E,\theta_V,t}(s_t, a_t) && (\forall\, 0 \le t \le T-1) \\
Q^{soft}_{\theta_E,\theta_V,t}(s_t, a_t) &= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\
&\quad + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \left[ V^{soft}_{\theta_E,\theta_V,t+1}(S_{t+1}) \,\middle|\, s_t, a_t \right] && (\forall\, 0 \le t < T-1) \\
Q^{soft}_{\theta_E,\theta_V,T-1}(s_{T-1}, a_{T-1}) &= \langle \theta_E \,,\, \phi(s_{T-1}, a_{T-1}) \rangle + \gamma^{T-1} \langle \theta_V \cdot \phi(s_{T-1}, a_{T-1}) \,,\, \phi(s_{T-1}, a_{T-1}) \rangle
\end{aligned} \tag{4.45}$$

*Proof.* Similar to [3] we begin by setting $\nabla_{\pi_t} \Psi$ to zero and rearrange the equation to find the form of the policy at optimality as the following

$$\begin{aligned}
&\pi_t(a_t \mid s_t) = \\
&\exp\Bigg( \langle \theta_E \,,\, \phi(s_t, a_t) \rangle - 1 + \frac{\mu_{s_t}}{\rho_{\pi_t,t}(s_t)} + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\
&+ \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \Big( \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle + \gamma^{t'} \langle \theta_V \cdot \phi(S_{t'}, A_{t'}) \,,\, \phi(S_{t'}, A_{t'}) \rangle - \log \pi_{t'}(A_{t'} \mid S_{t'}) \Big) \,\middle|\, s_t, a_t \right] \\
&+ 2 \cdot \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{i=t}^{T-1} \sum_{j=0}^{i-1} \gamma^{i+j-t} \langle \theta_V \cdot \phi(S_i, A_i) \,,\, \phi(S_j, A_j) \rangle \,\middle|\, s_t, a_t \right] \Bigg)
\end{aligned} \tag{4.46}$$

Setting $t = T-1$ we can make use of the notation with the assumption that the first expectation term vanishes to get

$$\begin{aligned}\pi_{T-1}(a_{T-1} \mid s_{T-1}) = \exp\Bigg( &\langle \theta_E \,,\, \phi(s_{T-1}, a_{T-1}) \rangle - 1 + \frac{\mu_{s_{T-1}}}{\rho_{\pi_{T-1},T-1}(s_{T-1})} \\ &+ \gamma^{T-1} \langle \theta_V \cdot \phi(s_{T-1}, a_{T-1}) \,,\, \phi(s_{T-1}, a_{T-1}) \rangle \\ &+ 2 \cdot \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{j=0}^{T-2} \gamma^{j+T-1-t} \langle \theta_V \cdot \phi(S_{T-1}, A_{T-1}) \,,\, \phi(S_j, A_j) \rangle \,\middle|\, s_{T-1}, a_{T-1} \right] \Bigg)\end{aligned} \tag{4.47}$$

As stated in [3], calculating the optimal policy from the given equations Eq. 4.46 and Eq. 4.47 would lead to an enumeration of exponentially many trajectories even just to approximate the expectation terms. A different approach however is to solve the problem via soft value iteration. For this we have to decompose Eq. 4.46 into some action value function $Q^{soft}$ and state value function $V^{soft}$ which can be done in the following way such that it will fit the form of our desired policy

$$V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) := 1 - \frac{\mu_{s_t}}{\rho_{\pi_t,t}(s_t)} - 2 \cdot \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{i=t}^{T-1} \sum_{j=0}^{i-1} \gamma^{i+j-t} \langle \theta_V \cdot \phi(S_i, A_i) \,,\, \phi(S_j, A_j) \rangle \,\middle|\, s_t, a_t \right] \tag{4.48}$$

$$\begin{aligned}Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) := &\langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\ &+ \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \big( \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle + \gamma^{t'} \langle \theta_V \cdot \phi(S_{t'}, A_{t'}) \,,\, \phi(S_{t'}, A_{t'}) \rangle \right. \\ &\left. - \log \pi_{t'}(A_{t'} \mid S_{t'}) \big) \,\middle|\, s_t, a_t \right]\end{aligned} \tag{4.49}$$

$$Q^{\text{soft}}_{\theta_E,\theta_V,T-1}(s_{T-1}, a_{T-1}) := \langle \theta_E \,,\, \phi(s_{T-1}, a_{T-1}) \rangle + \gamma^{T-1} \langle \theta_V \cdot \phi(s_{T-1}, a_{T-1}) \,,\, \phi(s_{T-1}, a_{T-1}) \rangle \tag{4.50}$$

As the authors motivate in [3], we can set the $\mu_{s_t}$ to appropriate normalizing constants such that the normalization KKT conditions in Eq. 4.28 will be satisfied. Further, as we only want to consider normalized policies $\pi_t \in \mathscr{P}$ we enforce the assumption that the $\mu_{s_t}$ will be chosen such that $\sum_{a \in \mathcal{A}} \pi_t(a \mid s_t) = 1$ holds for all $s_t \in \mathcal{S}$. This way we can argue exactly as in [3] that $V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)$ must be a soft maximum over $Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t)$ values in $s_t$.

$$1 = \sum_{a_t \in \mathcal{A}} \pi_{\theta_E,\theta_V,t}(a_t \,|\, s_t) \tag{4.51}$$

$$= \sum_{a_t \in \mathcal{A}} \exp(Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) - V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)) \tag{4.52}$$

$$1 \cdot \exp V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) = \left( \sum_{a_t \in \mathcal{A}} \exp(Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) - V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)) \right) \cdot \exp V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) \tag{4.53}$$

$$= \sum_{a_t \in \mathcal{A}} \exp Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) \tag{4.54}$$

$$V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) = \log \sum_{a_t \in \mathcal{A}} \exp Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) \tag{4.55}$$

Note that the derivations above are taken exactly from [3], only adapting the notation to the one chosen in this work. Especially $\pi_{\theta_E,\theta_V,t}$ denotes the time-dependent policy that depends on the parameters $\theta_e$ and $\theta_V$ and should coincide with the policy that satisfies Eq. 4.44. Next, we use the definitions that we have picked above for $Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t)$ and $V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)$ to see for $0 \leq t < T-1$

$$\begin{aligned} Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) &= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\ &\quad + \mathop{\mathbb{E}}_{\pi_t} \Bigg[ \sum_{t'=t+1}^{T-1} \gamma^{t'-t} \big( \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle + \gamma^{t'} \langle \theta_V \cdot \phi(S_{t'}, A_{t'}) \,,\, \phi(S_{t'}, A_{t'}) \rangle \\ &\quad - \log \pi_{t'}(A_{t'} \,|\, S_{t'}) \big) \,\Big|\, s_t, a_t \Bigg] \end{aligned} \tag{4.56}$$

$$\begin{aligned} &= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\ &\quad + \mathop{\mathbb{E}}_{\mathcal{T}} \Bigg[ \mathop{\mathbb{E}}_{\pi_t} \Bigg[ \gamma \big( \langle \theta_E \,,\, \phi(S_{t+1}, A_{t+1}) \rangle + \gamma^{t+1} \langle \theta_V \cdot \phi(S_{t+1}, A_{t+1}) \,,\, \phi(S_{t+1}, A_{t+1}) \rangle \\ &\quad - \log \pi_{t+1}(A_{t+1} \,|\, S_{t+1}) \big) \\ &\quad + \sum_{t'=t+2}^{T-1} \gamma^{t'-t} \big( \langle \theta_E \,,\, \phi(S_{t'}, A_{t'}) \rangle + \gamma^{t'} \langle \theta_V \cdot \phi(S_{t'}, A_{t'}) \,,\, \phi(S_{t'}, A_{t'}) \rangle \\ &\quad - \log \pi_{t'}(A_{t'} \,|\, S_{t'}) \big) \,\Big|\, S_{t+1} \Bigg] \,\Big|\, s_t, a_t \Bigg] \end{aligned} \tag{4.57}$$

$$\begin{aligned} &= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\ &\quad + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \Bigg[ \mathop{\mathbb{E}}_{\pi_t} \Bigg[ Q^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1}, A_{t+1}) - \log \pi_{t+1}(A_{t+1} \,|\, S_{t+1}) \,\Big|\, S_{t+1} \Bigg] \,\Big|\, s_t, a_t \Bigg] \end{aligned} \tag{4.58}$$

$$\begin{aligned} &= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\ &\quad + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \Bigg[ \mathop{\mathbb{E}}_{\pi_t} \Bigg[ Q^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1}, A_{t+1}) - \big( Q^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1}, A_{t+1}) \end{aligned}$$

$$- V^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1})) \,\Big|\, S_{t+1} \Big] \,\Big|\, s_t, a_t \Big] \tag{4.59}$$

$$= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \Big[ V^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1}) \,\Big|\, s_t, a_t \Big] \tag{4.60}$$

In total, we derived the following soft value iteration equations

$$\begin{aligned}
V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) &= \log \sum_{a_t \in \mathcal{A}} \exp Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) && (\forall\, 0 \le t \le T-1) \\
Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) &= \langle \theta_E \,,\, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \,,\, \phi(s_t, a_t) \rangle \\
&\quad + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \big[ V^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1}) \,\big|\, s_t, a_t \big] && (\forall\, 0 \le t < T-1) \\
Q^{\text{soft}}_{\theta_E,\theta_V,T-1}(s_{T-1}, a_{T-1}) &= \langle \theta_E \,,\, \phi(s_{T-1}, a_{T-1}) \rangle + \gamma^{T-1} \langle \theta_V \cdot \phi(s_{T-1}, a_{T-1}) \,,\, \phi(s_{T-1}, a_{T-1}) \rangle
\end{aligned} \tag{4.61}$$

□

Note that the soft aspect of the soft value iteration process is the fact that we use a *log-sum-exp* instead of a hard maximum, as it is used in traditional value iteration.

## 4.3 Dual Ascent Algorithm

As we have derived a similar solution to the Optimization Problem 1 as for the one using only matching feature expectations, we also use an algorithm that is based on the same approach and just swap out the soft value iteration and update the additional Lagrange multiplier that corresponds to the variance matching.

The algorithm in [3] is based on dual ascent. The policy $\pi_t$ that will attain the value of the dual function $g(\theta_E, \theta_V) = \Lambda(\pi_t, \theta_E, \theta_V)$ can be found by using the soft value iteration that we have derived in the previous section. For updating the dual variables $\theta_E$ and $\theta_V$ we look at the gradients of $\Lambda(\pi_t, \theta_E, \theta_V)$ which are

$$\nabla_{\theta_E} \Lambda(\pi_t, \theta_E, \theta_V) = \mathop{\mathbb{E}}_{\pi_t} \Big[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big] - \mathop{\mathbb{E}}_{\mathcal{D}} \Big[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big] \tag{4.62}$$

$$\begin{aligned}
\nabla_{\theta_V} \Lambda(\pi_t, \theta_E, \theta_V) &= \mathop{\mathbb{E}}_{\pi_t} \Big[ \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big) \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big)^T \Big] \\
&\quad - \mathop{\mathbb{E}}_{\mathcal{D}} \Big[ \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big) \Big( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \Big)^T \Big]
\end{aligned} \tag{4.63}$$

The expectation terms regarding $\pi_t$ can be obtained by applying soft value iteration to $\theta_E$, $\theta_V$ and then rolling the optimal policy forward to obtain occupancy measures as mentioned in [3]. The expectation terms regarding the demonstrator $\mathcal{D}$ are easily

computed from the demonstration trajectories. The whole algorithm for updating $\pi_t$ and $\theta_E$, $\theta_V$ can be found in Algorithm 1.

**Algorithm 1** Maximum Causal Entropy (MCE) IRL on demonstrator $\mathcal{D}$ using feature variance matching to solve the dual of Optimization Problem 1

**Input**: RL Environment for computing feature mappings, demonstrator trajectories to compute the expectation values w.r.t. $\mathcal{D}$, agent implementing soft value iteration for given $\theta_E$, $\theta_V$ to compute $\pi_{\theta_E,\theta_V,t}$
Initialize some parameter estimates $\theta_E^0$, $\theta_V^0$
$k = 0$
**while** Stopping condition not satisfied **do**
Apply soft VI to obtain optimal policy $\pi^k_{\theta_E,\theta_V,t}$ w.r.t. $\theta_E^k$, $\theta_V^k$ (Eq. 4.45)
$\theta_E^{k+1} = \theta_E^k + \alpha_E^k \left( \mathbb{E}_{\mathcal{D}} \left[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right] - \mathbb{E}_{\pi^k_{\theta_E,\theta_V,t}} \left[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right] \right)$ (Eq. 4.62)
$\theta_V^{k+1} = \theta_V^k + \alpha_V^k \left( \mathbb{E}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right) \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^T \right] - \mathbb{E}_{\pi^k_{\theta_E,\theta_V,t}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right) \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^T \right] \right)$ (Eq. 4.63)
$k = k + 1$
**end while**
**Returns**: $\pi^k_{\theta_E,\theta_V,t}$, $\theta_E^k$, $\theta_V^k$

## 4.4 Issues for Infinite-horizon MDPs

Throughout the derivations of the previous sections, we've worked under the assumption that our MDP has a finite horizon. This implies that we can employ the soft value iteration by simply applying the equations recursively from $T-1$ down to $t = 0$ in order to compute $Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t)$ and $V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)$ and consequently derive the policy $\pi_{\theta_E,\theta_V,t}(a_t \,|\, s_t)$ for every time step $t$ and state $s_t$ [3].

The original approach with solely feature matching is able to generalize the algorithm to stationary policies and especially infinite-horizon MDPs. The issue with the approach of this work is that in the soft value iteration we have added a term that explicitly depends on time step $t$, namely the factor $\gamma^t$ in front of the newly incorporated term $\langle \theta_V \cdot \phi(s_t, a_t)\,,\, \phi(s_t, a_t) \rangle$. In order to lose this dependency, we would have to ensure $\gamma = 1$, rendering the approach inapplicable for infinite-horizon MDPs. Addressing this challenge and adapting the approach to such MDPs could be a focal point for future research, which remains unexplored in this thesis.

## 4.5 Solution for LQRs

In this section we develop a closed form solution for the special case of LQRs. Note that for this section we assume that the feature representation of state-action pairs will just be the state itself to be consistent with the results in [4] that only consider the cost/reward function w.r.t. state variables.

**Theorem 3.** *For the special case where dynamics are linear functions with Gaussian noise, the extended MCE IRL model permits a closed form solution and, given dynamics* $s_{t+1} \sim N(As_t + Ba_t, \Sigma)$*, Eq. 2.4 reduces to:*

$$Q^{soft}_{\theta_E,\theta_V,t}(s_t, a_t) = s_t^T R + \gamma^t s_t^T Q s_t + \gamma \left[ \begin{pmatrix} a_t \\ s_t \end{pmatrix}^T \begin{pmatrix} B^T D B & A^T D B \\ B^T D A & A^T D A \end{pmatrix} \begin{pmatrix} a_t \\ s_t \end{pmatrix} + \begin{pmatrix} a_t \\ s_t \end{pmatrix}^T \begin{pmatrix} B^T G \\ A^T G \end{pmatrix} \right] \tag{4.64}$$

$$V^{soft}_{\theta_E,\theta_V,t}(s_t) = s_t^T (R + \gamma F_s) + s_t^T \left( \gamma^t Q - \gamma C_{as}^T C_{aa}^{-1} C_{as} + \gamma C_{ss} \right) s_t \tag{4.65}$$

*where* $C$ *and* $D$ *are recursively computed as:* $C_{aa} = B^T D B$, $C_{sa} = C_{as}^T = B^T D A$, $C_{ss} = A^T D A$, $D = \gamma C_{ss} + \gamma^t Q - \gamma C_{as}^T C - aa^{-1} C_{as}$, $G = \gamma F_s + R$.

*Proof.* We are going to prove this theorem very similar to [4, Theorem 6.15] as it is the extension of this theorem for our approach of also matching feature variances while also assuming the presence of a discount factor $\gamma$. Using the matrices $Q$ and $R$ from the definition of the cost function as Lagrange multipliers and applying soft value iteration defined in Lemma 2 to this setting we get the following equations that we need to solve.

$$Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t)(s_t, a_t) = \langle R\,,\, s_t \rangle + \gamma^T \langle Q \cdot s_t\,,\, s_t \rangle + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \left[ V^{\text{soft}}_{\theta_E,\theta_V,t+1}(S_{t+1}) \,\middle|\, s_t, a_t \right] \tag{4.66}$$

$$V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) = \log \sum_{a_t \in \mathcal{A}} \exp Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t)(s_t, a_t) \tag{4.67}$$

Similar to [4] we will first assume the quadratic form of $Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t)$ and $V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)$ and then verify that this definition will solve the equations above. More specifically, we will assume that they obtain the following form

$$Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t, a_t) = s_t^T R + \gamma^t s_t^T Q s_t + \gamma \left[ \begin{pmatrix} a_t \\ s_t \end{pmatrix}^T \begin{pmatrix} C_{aa} & C_{as} \\ C_{sa} & C_{aa} \end{pmatrix} \begin{pmatrix} a_t \\ s_t \end{pmatrix} + \begin{pmatrix} a_t \\ s_t \end{pmatrix}^T \begin{pmatrix} F_a \\ F_s \end{pmatrix} \right] \tag{4.68}$$

$$V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) = s_t^T G + s_t^T D s_t \tag{4.69}$$

Based on this, we can express Eq. 4.66 further as

$$\begin{aligned}
Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t,a_t) &= \langle R\,,\,s_t\rangle + \gamma^t\langle Q\cdot s_t\,,\,s_t\rangle + \gamma\,\mathop{\mathbb{E}}_{\mathcal{T}}\Big[\langle G\,,\,s_t\rangle + \langle D\cdot s_t\,,\,s_t\rangle \,\Big|\, s_t,a_t\Big] \\
&= s_t^T R + \gamma^t s_t^T Q s_t \\
&\quad + \gamma\Big[(As_t+Ba_t)^T D(As_t+Ba_t) + tr(D\Sigma) + a_t^T B^T G + s_t^T A^T G\Big] \\
&= s_t^T R + \gamma^t s_t^T Q s_t \\
&\quad + \gamma\left[\begin{pmatrix}a_t\\ s_t\end{pmatrix}^T \begin{pmatrix}B^TDB & A^TDB\\ B^TDA & A^TDA\end{pmatrix}\begin{pmatrix}a_t\\ s_t\end{pmatrix} + \begin{pmatrix}a_t\\ s_t\end{pmatrix}^T \begin{pmatrix}B^TG\\ A^TG\end{pmatrix}\right] + const
\end{aligned} \tag{4.70}$$

Note that above we made use of $\langle A\,,\,B\rangle = B^TA$ for vectors/column matrices and the symmetry of $Q$.

Given those reformulations, we can define the following update rules: $C_{aa} = B^TDB$, $C_{sa} = C_{as}^T = B^TDA$, $C_{ss} = A^TDA$, $F_a = B^TG$, $F_s = A^TG$.

We further also reformulate $V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t)$ based on this and Eq. 4.67

$$\begin{aligned}
&V^{\text{soft}}_{\theta_E,\theta_V,t}(s_t) = \\
&= \log\sum_{a_t\in\mathcal{A}}\exp\left(s_t^TR + \gamma^t s_t^TQs_t + \gamma\big(a_t^TC_{aa}a_t + a_t^TC_{as}s_t + s_t^T\underbrace{C_{sa}}_{=C_{as}^T}a_t + s_t^TC_{ss}s_t + s_t^TF_s + a_t^TF_a\big)\right) \\
&= s_t^TR + \gamma^Ts_t^TQs_t + \gamma s_t^TC_{ss}s_t + \gamma s_t^TF_s \\
&\quad + \log\sum_{a_t\in\mathcal{A}}\exp\left(\gamma\big((a_t^TC_{aa}a_t + a_t^TC_{as}s_t + s_t^T\underbrace{C_{sa}}_{=C_{as}^T}a_t + a_t^TF_a\big)\right) \\
&= s_t^TR + \gamma^Ts_t^TQs_t + \gamma s_t^TC_{ss}s_t + \gamma s_t^TF_s \\
&\quad + \log\sum_{a_t\in\mathcal{A}}\exp\left(\gamma\big((a_t+C_{aa}^{-1}C_{as}s_t)^TC_{aa}(a_t+C_{aa}^{-1}C_{as}s_t) - s_t^TC_{as}^TC_{aa}^{-1^T}C_{aa}C_{aa}^{-1}C_{as}s_t\big)\right) \\
&= s_t^TR + \gamma^Ts_t^TQs_t + \gamma s_t^TC_{ss}s_t + \gamma s_t^TF_s - \gamma s_t^TC_{as}^TC_{aa}^{-1^T}C_{aa}C_{aa}^{-1}C_{as}s_t + const \\
&= s_t^T(R+\gamma F_s) + s_t^T\big(\gamma^tQ + \gamma C_{ss} - \gamma C_{as}^TC_{aa}^{-1^T}C_{aa}C_{aa}^{-1}C_{as}\big)s_t
\end{aligned} \tag{4.71}$$

which gives use the update rules: $D = \gamma^tQ + \gamma C_{ss} - \gamma C_{as}^TC_{aa}^{-1^T}C_{aa}C_{aa}^{-1}C_{as}$, $G = F_s + R$.

The stochastic policy is then $\pi_t(a_t\,|\,s_t) \propto e^{Q^{\text{soft}}_{\theta_E,\theta_V,t}(s_t,a_t)}$. □

## 4.6 Generalization for Skewness and Higher-order Moments

As a concluding aspect of the theoretical contributions in this work, it's worth considering not only the feature expectation and variance but also its skewness and higher-order moments. Hence, we aim to demonstrate briefly that all definitions and proofs can be extended to encompass arbitrary higher-order moments, provided that all moments up to a certain order $n$ are matched.

### 4.6.1 Problem Definition

In order to match the feature skewness, we make use of the following property that holds for finite expectation and variance, which we can assume to hold in our setting.

$$\begin{aligned}\mathbb{E}\Big[(X - \mathbb{E}\big[X\big])^3\Big] &= \mathbb{E}\big[X^3\big] - 3\,\mathbb{E}\big[X\big]\,\mathbb{E}\big[X^2\big] + 3\,\mathbb{E}\big[X\big]^2\,\mathbb{E}\big[X\big] - \mathbb{E}\big[X\big]^3 \\ &= \mathbb{E}\big[X^3\big] - 3\,\mathbb{E}\big[X\big]\big(\mathbb{E}\big[X^2\big] + \mathbb{E}\big[X\big]\,\mathbb{E}\big[X\big] - \mathbb{E}\big[X\big]^3 \\ &= \mathbb{E}\big[X^3\big] - 3\,\mathbb{E}\big[X\big]\,\mathbb{V}\big[X\big] - \mathbb{E}\big[X\big]^3 \end{aligned} \tag{4.72}$$

We see then, similar as we have shown as motivation for feature variance matching, that we only need to add the matching of the $\mathbb{E}\big[X^3\big]$ term. So we can add the constraint in Eq. 4.73 to our Optimization Problem 1 to also consider matching the feature skewness.

$$\mathop{\mathbb{E}}_{\pi_t}\left[\left(\sum_{t=0}^{T-1}\gamma^t\phi(S_t, A_t)\right)^{\otimes 3}\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\left(\sum_{t=0}^{T-1}\gamma^t\phi(S_t, A_t)\right)^{\otimes 3}\right], \tag{4.73}$$

where $\cdot^{\otimes 3}$ denotes the outer product of the input three times with itself. Thus, the expectation values in Eq. 4.73 are in $\mathbb{R}^{d\times d\times d}$.

We can also see that this can be extended to the general case of matching the first $n$ moments for some $n \geq 2$ as

$$\mathbb{E}\Big[(X - \mathbb{E}\big[X\big])^n\Big] = \sum_{k=0}^{n}\binom{n}{k}\mathbb{E}\big[X^k\big](-\mathbb{E}\big[X\big]^{n-k}) = \mathbb{E}\big[X^n\big] + \sum_{k=0}^{n-1}\binom{n}{k}\mathbb{E}\big[X^k\big](-\mathbb{E}\big[X\big]^{n-k}) \tag{4.74}$$

Thus, we always only have to consider the $\mathbb{E}\big[X^n\big]$ for matching the $n$-th moment, provided we are already doing that for $0 \leq k < n$ as well. Thus, we only have to add the following constraints to the existing problem definition

$$\mathop{\mathbb{E}}_{\pi_t}\left[\left(\sum_{t=0}^{T-1}\gamma^t\phi(S_t, A_t)\right)^{\otimes n}\right] = \mathop{\mathbb{E}}_{\mathcal{D}}\left[\left(\sum_{t=0}^{T-1}\gamma^t\phi(S_t, A_t)\right)^{\otimes n}\right] \tag{4.75}$$

### 4.6.2 Algorithm Derivation

As we have seen in the previous section, in order to add the skewness matching or in general the $n$-th moment matching we get additional constraints of the form in Eq. 4.75.

Thus, when we formulate the Lagrangian for this extended problem, we introduce an additional parameter $\theta_{M^k} \in \mathbb{R}^{\times^k d}$, where $\times^k d$ denotes $\underbrace{d \times \cdots \times d}_{\text{k times}}$, for the constraint of the $k$th - moment with $3 \leq k \leq n$ and the Lagrangian gets the additional summands

$$\langle \theta_{M^k} \, , \, \mathop{\mathbb{E}}_{\pi_t} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes k} \right] - \mathop{\mathbb{E}}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes k} \right] \rangle \tag{4.76}$$

Note, that again we assume a property of symmetry for these parameters $\theta_{M^k}$. As they are $k$-dimensional tensors we consider here a symmetry in all dimensions, i.e., an invariance regarding index permutations. This will help us to simplify the gradient, similar to the case of variance matching.

That further means that we also need the derivative $\nabla_{\pi_t}$ of this term. Here we can basically make the same observations as for the second moment. We will show a shortened derivation for the case of the third moment that uses similar modifications as the derivation for the variance.

$$\begin{aligned}
&\nabla_{\pi_t(a_t \,|\, s_t)} \langle \theta_{M^3} \, , \, \mathop{\mathbb{E}}_{\pi_t} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes 3} \right] - \mathop{\mathbb{E}}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes 3} \right] \rangle \\
&\quad = \nabla_{\pi_t(a_t \,|\, s_t)} \langle \theta_{M^3} \, , \, \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{i,j,k=0}^{T-1} \gamma^{i+j+k} \underbrace{\phi(S_i, A_i) \otimes \phi(S_j, A_j) \otimes \phi(S_k, A_k)}_{\Phi_{ijk}} \right] \rangle \\
&\quad = \rho_{\pi_t, t}(s_t) \langle \theta_{M^3} \, , \, \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{\substack{i,j,k=0 \\ i \geq t \vee j \geq t \vee k \geq t}}^{T-1} \gamma^{i+j+k-t} \Phi_{ijk} \,\middle|\, s_t, a_t \right] \rangle \\
&\quad = \rho_{\pi_t, t}(s_t) \Bigg( \gamma^{2t} \langle \theta_{M^3} \, , \, \Phi_{ttt} \rangle + \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{t'=t+1}^{T-1} \gamma^{3t'-t} \langle \theta_{M^3} \, , \, \Phi_{t't't'} \rangle \,\middle|\, s_t, a_t \right] \\
&\qquad + 3 \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{i=t}^{T-1} \sum_{j=0}^{i} \sum_{k=0}^{i-1} \gamma^{i+j+k-t} \langle \theta_{M^3} \, , \, \Phi_{ijk} \rangle \,\middle|\, s_t, a_t \right] \Bigg)
\end{aligned} \tag{4.77}$$

Note, that we were able to simplify the triple sum in this way due to the symmetry of $\theta_{M^k}$.

This can be extended W.L.O.G. to the higher order moments as well, due to the properties of the Frobenius inner product and the outer product of tensors. We will only state the final derivation for the general case here.

$$\begin{aligned}
&\nabla_{\pi_t(a_t \,|\, s_t)} \langle \theta_{M^k} \, , \, \mathop{\mathbb{E}}_{\pi_t} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes k} \right] - \mathop{\mathbb{E}}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes k} \right] \rangle \\
&\quad = \rho_{\pi_t, t}(s_t) \Bigg( \gamma^{(k-1)t} \langle \theta_{M^k} \, , \, \Phi_{t^k} \rangle + \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{t'=t+1}^{T-1} \gamma^{kt'-t} \langle \theta_{M^k} \, , \, \Phi_{t'^k} \rangle \,\middle|\, s_t, a_t \right] \\
&\qquad + k \mathop{\mathbb{E}}_{\pi_t} \left[ \sum_{j_1=t}^{T-1} \sum_{j_2=0}^{j_1} \cdots \sum_{j_{k-1}=0}^{j_1} \sum_{j_k=0}^{j_1 - 1} \gamma^{\sum j_i - t} \langle \theta_{M^k} \, , \, \Phi_J \rangle \,\middle|\, s_t, a_t \right] \Bigg)
\end{aligned} \tag{4.78}$$

where $\Phi_{t^k} = \underbrace{\phi(S_t, A_t) \otimes \cdots \otimes \phi(S_t, A_t)}_{\text{k times}}$ and $\Phi_J = \phi(S_{j_1}, A_{j_1}) \otimes \cdots \otimes \phi(S_{j_k}, A_{j_k})$.

We can then use these additional terms to modify the soft value iteration that we defined in Eq. 4.45. We will already state the extension to the general case of matching the first $n$ moments.

**Lemma 4.** *The KKT condition* $\nabla_{\pi_t} \Psi(\pi_t, \mu; \theta_E, \theta_V, \theta_{M^3}, \ldots, \theta_{M^n}) = 0$ *is satisfied by a policy*

$$\pi_t(a_t \,|\, s_t) = \exp\left( Q^{soft}_{\Theta,t}(s_t, a_t) - V^{soft}_{\Theta,t}(s_t) \right) \tag{4.79}$$

*where* $\Theta$ *denotes the dependence on all parameters* $\theta_E, \theta_V, \theta_{M^3}, \ldots, \theta_{M^n}$*. This policy satisfies the following recursion:*

$$\begin{aligned}
V^{soft}_{\Theta,t}(s_t) &= log \sum_{a_t \in \mathcal{A}} \exp Q^{soft}_{\Theta,t}(s_t, a_t) && (\forall 0 \le t \le T-1) \\
Q^{soft}_{\Theta,t}(s_t, a_t) &= \langle \theta_E \, \phi(s_t, a_t) \rangle + \gamma^t \langle \theta_V \cdot \phi(s_t, a_t) \, , \, \phi(s_t, a_t) \rangle \\
&\quad + \sum_{k=3}^{n} \gamma^{(k-1)t} \langle \theta_{M^k} \, , \, \Phi_{t^k} \rangle + \gamma \mathop{\mathbb{E}}_{\mathcal{T}} \left[ V^{soft}_{\Theta,t+1}(S_{t+1}) \,\middle|\, s_t, a_t \right] \quad (\forall 0 \le t < T-1) \\
Q^{soft}_{\Theta,T-1}(s_{T-1}, a_{T-1}) &= \langle \theta_E \, \phi(s_{T-1}, a_{T-1}) \rangle + \gamma^{T-1} \langle \theta_V \cdot \phi(s_{T-1}, a_{T-1}) \, , \, \phi(s_{T-1}, a_{T-1}) \rangle \\
&\quad + \sum_{k=3}^{n} \gamma^{(k-1)(T-1)} \langle \theta_{M^k} \, , \, \Phi_{(T-1)^k} \rangle
\end{aligned} \tag{4.80}$$

*Proof.* One can use exactly the same approach as in the proof of Lemma 2. □

Collecting all parts together, we finally obtain the modified algorithm to also consider higher order moments that is given in Algorithm 2.

**Algorithm 2** MCE IRL on demonstrator $\mathcal{D}$ using higher order moment matching to solve the dual of Optimization Problem 1 with additional constraints for higher order moment matching

**Input**: RL Environment for computing feature mappings, demonstrator trajectories to compute the expectation values w.r.t. $\mathcal{D}$, agent implementing soft value iteration for given $\theta_E$, $\theta_V, \theta^k_{M^3}, \ldots, \theta^k_{M^n}$ to compute $\pi^k_{\Theta,t}$
Initialize some parameter estimates $\theta^0_E$, $\theta^0_V, \theta^0_{M^3}, \ldots, \theta^0_{M^n}$
$k = 0$
**while** Stopping condition not satisfied **do**
Apply soft VI to obtain optimal policy $\pi^k_{\Theta,t}$ w.r.t. $\Theta^k$ (Eq. 4.80)

$$\theta^{k+1}_E = \theta^k_E + \alpha^k_E \left( \mathbb{E}_{\mathcal{D}} \left[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right] - \mathbb{E}_{\pi^k_{\Theta,t}} \left[ \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right] \right) \text{ (Eq. 4.62)}$$

$$\theta^{k+1}_V = \theta^k_V + \alpha^k_V \left( \mathbb{E}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes 2} \right] - \mathbb{E}_{\pi^k_{\Theta,t}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes 2} \right] \right) \text{ (Eq. 4.63)}$$

$$\theta^{k+1}_{M^3} = \theta^k_{M^3} + \alpha^k_3 \left( \mathbb{E}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes 3} \right] - \mathbb{E}_{\pi^k_{\Theta,t}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes 3} \right] \right)$$

$$\vdots$$

$$\theta^{k+1}_{M^n} = \theta^k_{M^n} + \alpha^k_n \left( \mathbb{E}_{\mathcal{D}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes n} \right] - \mathbb{E}_{\pi^k_{\Theta,t}} \left[ \left( \sum_{t=0}^{T-1} \gamma^t \phi(S_t, A_t) \right)^{\otimes n} \right] \right)$$

$k = k + 1$
**end while**
**Returns**: $\pi^k_{\Theta,t}$, $\theta^k_E$, $\theta^k_V, \theta^k_{M^3}, \ldots, \theta^k_{M^n}$

# 5 Experiments

In this chapter, we present a proof-of-concept example illustrating the potential usefulness of our new approach compared to the original one. The code for this experiment is available at Maximum Causal Entropy IRL. We will demonstrate scenarios where additionally matching feature variances can lead to better alignment with the demonstrator's behavior. To maintain simplicity, we focus on a finite horizon example with $\gamma$ set to 1.

## 5.1 Problem Setup

For our experiment, we use a grid world scenario. The agent starts in a specified starting state on the grid and should move towards one of the specified end states as fast as possible while simultaneously picking up items placed around the grid. These items have different feature vectors.

Figure 5.1 displays the specific grid configuration for this experiment. The green cell represents the agent's starting state, and the target states are colored orange. In each of the states, the agent can pick one of the following actions: move up, down, left or right, as long as the action does not lead outside of the grid. The actions are deterministic and lead with 100% probability to the expected state. Once the agent reaches one of the target states, it cannot chose any further actions and the agent reaches an absorbing state that it cannot leave.

We observe that depending on the features mappings of the states containing the different items, any optimal policy should follow one of the three paths depicted in Figure 5.2 with some probability and no other paths.

### Feature Mappings and Demonstrator

For our feature mappings, we only consider state-dependent features, i.e., $\phi(s_t, a_t) = \phi(s_t) \in \mathbb{R}^3$, which are defined as follows:

1. for empty states: $[0, 0, 1]$
2. for the state with $\diamond$: $[1, 1, 1]$
3. for the state with $\triangle$: $[2, 0, 1]$
4. for the state with $\square$: $[0, 2, 1]$

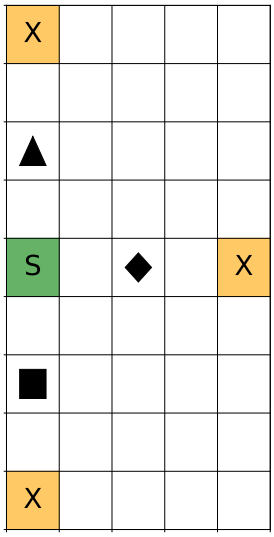


Figure 5.1: Grid world set up for the experiment. The starting state is green, the target states are orange. The diamond, triangle and square are the items that can be collected.

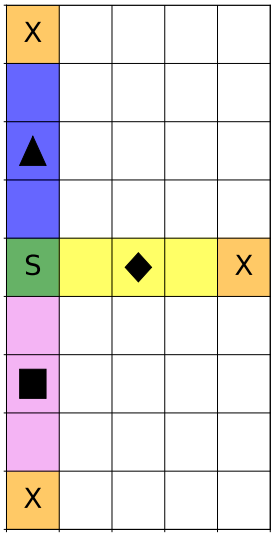


Figure 5.2: Three optimal paths (blue, yellow, pink) in the grid world.

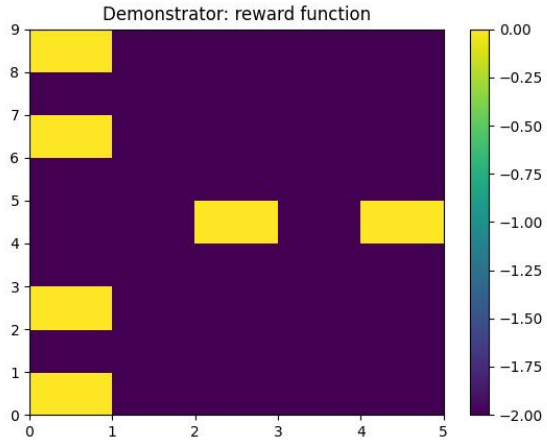


Figure 5.3: Reward function of the demonstrator for the different states.

5. for target states: $[1, 1, 1]$.

Together with the true reward parameters set to $[1, 1, -2]$, the first two features ensure that the agent is positively rewarded for picking up items and reaching one of the target states. The third feature will count the steps the agent takes while moving around the grid. The negative reward for this feature ensures that the agent will try to minimize its number of steps to reach a target state. The reward function with parameters $\theta_\star = [1, 1, -2]$ is shown in Figure 5.3. Note that with the feature mapping defined as above, the agent does not get a higher reward for reaching a target state than for picking up an item. We will later see that this is not an issue due to the manual definition of the demonstrator's policy. In future experiments, this should be adapted to observe if there are any changes in behavior or convergence.

For our experiment, we manually define the behavior of the demonstrator to take either the blue or the pink path with equal probability, i.e., it either goes up or down with 50% probability each. Thus, our demonstrator will completely ignore the yellow path.

### Different Policies, Different Variances

In the setting we defined, multiple policies can achieve the same feature expectations but have different feature variances. We consider the only possible optimal paths depicted in Figure 5.2.

Given the feature mappings that we defined above and the fact that $\gamma = 1$, the following three policies have the same feature expectation count, namely $[2, 2, 5]$:

1. Policy $\pi_1$ that follows the blue or the pink path with equal probability, i.e., the demonstrator.

2. Policy $\pi_2$ that deterministically follows the yellow path.

3. Policy $\pi_3$ that follows any of the three paths with equal probability.

While these three policies share the same feature expectation, their variances are as follows:

$$\begin{aligned}
\underset{\pi_1}{\mathbb{V}}[\sum_{t=0}^{T-1}\phi(s_t)] &= \frac{1}{2}\left(\begin{bmatrix}3\\1\\5\end{bmatrix}-\begin{bmatrix}2\\2\\5\end{bmatrix}\right)^{\otimes 2}+\frac{1}{2}\left(\begin{bmatrix}1\\3\\5\end{bmatrix}-\begin{bmatrix}2\\2\\5\end{bmatrix}\right)^{\otimes 2}=\begin{bmatrix}1&-1&0\\-1&1&0\\0&0&0\end{bmatrix}\\
\underset{\pi_2}{\mathbb{V}}[\sum_{t=0}^{T-1}\phi(s_t)] &= \frac{1}{1}\left(\begin{bmatrix}2\\2\\5\end{bmatrix}-\begin{bmatrix}2\\2\\5\end{bmatrix}\right)^{\otimes 2}=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\end{bmatrix}\\
\underset{\pi_3}{\mathbb{V}}[\sum_{t=0}^{T-1}\phi(s_t)] &= \frac{1}{3}\left(\begin{bmatrix}3\\1\\5\end{bmatrix}-\begin{bmatrix}2\\2\\5\end{bmatrix}\right)^{\otimes 2}+\frac{1}{3}\left(\begin{bmatrix}1\\3\\5\end{bmatrix}-\begin{bmatrix}2\\2\\5\end{bmatrix}\right)^{\otimes 2}+\frac{1}{3}\left(\begin{bmatrix}2\\2\\5\end{bmatrix}-\begin{bmatrix}2\\2\\5\end{bmatrix}\right)^{\otimes 2}\\
&= \begin{bmatrix}\frac{2}{3}&-\frac{2}{3}&0\\-\frac{2}{3}&\frac{2}{3}&0\\0&0&0\end{bmatrix}
\end{aligned} \tag{5.1}$$

Since our approach uses the Maximum Causal Entropy principle, we expect any agent that only uses feature expectation matching to learn a policy similar to $\pi_3$, as it is the policy with the highest uncertainty. An agent that additionally uses feature variance matching should then be able to discern that policies $\pi_2$ and $\pi_3$ are not behaving as desired due to the different feature variances and thus learn a policy that behaves similarly to the demonstrator.

## 5.2 Results

This experiment was run on an Intel Core i9-13900H processor with 32 GB of RAM. The implementation was developed using Python 3.11.5 making use of mainly standard libraries like *numpy* for array manipulation, *scipy* for dealing with sparse transition matrices, *matplotlib* for visualization and *time* for runtime evaluations. No specific efforts have been put into the parallelization of the code. For further implementation details we refer to the GIT repository.

To run the experiment, we need the demonstrator's feature expectation and variance values. First, we manually compute the demonstrator's policy to generate trajectories in the given setting. Note that we only define a non-zero probability to move to different states on the left border of the grid, as we only care about defining the behavior of taking one of the two paths with equal probability. We do not concern ourselves with what happens in the other states, as those are never visited by the demonstrator's policy and thus do not appear in the state value counts.


(a) Policy of the demonstrator.

(b) Policy of Feature Expectation Matching

(c) Policy of Feature Variance Matching

Figure 5.4: The policies for different agents: Demonstrator manually defined only in the necessary states, an agent using feature expectation matching, and an agent using feature expectation and variance matching. The numbers in the state show the time step $t$ of the policy $\pi_t$ displayed in the different states.

| | Agent Expectation | Agent Variance |
|---|---|---|
| Reward (true: -6) | -6.000003 | -6.0 |
| # iterations | 5 | 1535 |
| Avg. time per iteration (sec.) | 0.02249 | 1.62926 |
| Total runtime (sec.) | 0.11249 | 2500.90669 |

Table 5.1: Numerical results from the experiment.

We then executed the MCE IRL algorithm with $T = 20$ to allow the agents to explore the grid in each episode, once using only feature expectation and once with additional feature variance matching. The constructed policies are shown in Figure 5.4.

Note that as we computed time dependent policies, Figure 5.4 shows the policy at time step $t$ for each state $s$ such that $s$ is reachable from the starting state $s_0$ in $t$ steps. This time step $t$ is displayed in the lower left corner of each state, showing the that the policy at time step $t$ is displayed for this specific state. The color of the state displays the value function $V(s_t)$, i.e., the expected future reward given the specific state.

As observed in Figure 5.4, the agent that only uses feature expectation matching learns the three paths with equal probability as anticipated. On the other hand, the agent utilizing feature variance matching appears to have learned that the two paths of the demonstrator are the desired behavior and thus follows those two paths with seemingly equal probability. Therefore, in terms of policy, the agent with variance matching seems to learn the behavior better, as expected.

In Table 5.1, we observe several numerical outcomes of the experiment. We note that, disregarding rounding errors, both agents achieve the reward of the demonstrator.

However, even though the agent with variance matching returns a policy with closer alignment to the demonstrator's behavior, it also leads to increased computational effort. In terms of iterations needed until convergence of the dual parameters, it requires considerably more iterations than using only feature expectation matching. Furthermore, the individual iterations also take considerably more time, by a factor of ~80. This results in a total runtime of more than 40 minutes, which is significantly longer than the not even one second required for the agent using only feature expectation matching.

This experiment demonstrates that using feature variance matching can achieve behavior closer to that of the demonstrator compared to using only feature expectation matching. Consequently, this approach can be beneficial in scenarios where replicating the demonstrator's behavior is crucial, and the increased computational complexity is acceptable.

# 6 Conclusion and Future Work

In this work, we have presented a new algorithm for Inverse Reinforcement Learning. We extended Ziebart et al.'s [2] Maximum Causal Entropy to include feature variance matching. This enhancement aims to achieve an even better approximation of the demonstrator's behavior and more expressive reward functions. We detailed all necessary steps and extended the proofs in [3] to establish a solid theoretical foundation for this new algorithm.

We demonstrated that for this new setting, we can still find a closed-form solution in the special case of Linear Quadratic Regulators. As a final step, we extended this approach to match arbitrary higher-order moments, motivated by the same goal as feature variance matching: achieving an even better approximation of the demonstrator's behavior. We provided proof ideas for this further developed approach and showed that they are a natural extension of the proofs for feature variance matching.

Furthermore, we noted that the new approach in this work is difficult to generalize to stationary policies and, therefore, to infinite-horizon Markov Decision Processes due to the explicit time dependence in the soft value iteration. Generalizing this idea to those settings might be an interesting topic for future work.

Lastly, we provided a proof-of-concept example in a grid world setting, demonstrating the potential strength of this work's algorithm. In the given scenario, our new algorithm achieved a behavior that is more similar to that of the demonstrator in comparison to only using feature expectation matching with the disadvantage of an increased computational effort. In future work, it would be interesting to design more intricate and complex scenarios to further test and compare the quality of feature variance matching with its precursor.

In conclusion, this work advances the field of Inverse Reinforcement Learning by incorporating feature variance matching into the existing framework. This enhancement helps better approximate demonstrator behavior. Future research can expand on these findings by exploring broader applications, refining the theory, and addressing challenges like generalizing to infinite-horizon settings. We believe these contributions will inspire further developments and lead to more robust algorithms capable of handling more complex real-world problems.

# Bibliography


[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[2] B. D. Ziebart, J. A. Bagnell, and A. K. Dey, "Modeling interaction via the principle of maximum causal entropy," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, 2010, p. 1255–1262.

[3] A. Gleave and S. Toyer, "A primer on maximum causal entropy inverse reinforcement learning," 2022.

[4] B. D. Ziebart, "Modeling purposeful adaptive behavior with the principle of maximum causal entropy," Ph.D. dissertation, USA, 2010, aAI3438449.

[5] A. Anwar, "Basic terminologies of reinforcement learning," https://medium.com/analytics-vidhya/basic-terminology-reinforcement-learning-2357fd5f0e51, accessed: 2024-06-25.

[6] C. Szepesvari, *Algorithms for Reinforcement Learning*. Morgan and Claypool Publishers, 2010.

[7] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. ICML '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 1. [Online]. Available: https://doi.org/10.1145/1015330.1015430

[8] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 663–670.

[9] M. Baert, P. Mazzaglia, S. Leroux, and P. Simoens, "Maximum causal entropy inverse constrained reinforcement learning," 2023. [Online]. Available: https://arxiv.org/abs/2305.02857

[10] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018. [Online]. Available: https://arxiv.org/abs/1801.01290

[11] T. V. Bui, T. Mai, and P. Jaillet, "Weighted maximum entropy inverse reinforcement learning," 2022.

[12] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum entropy deep inverse reinforcement learning," 2016.

[13] R. Shah, D. Krasheninnikov, J. Alexander, P. Abbeel, and A. Dragan, "Preferences implicit in the state of the world," 2019. [Online]. Available: https://arxiv.org/abs/1902.04198

[14] C. Finn, P. Christiano, P. Abbeel, and S. Levine, "A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models," 2016. [Online]. Available: https://arxiv.org/abs/1611.03852

[15] A. Y. Ng, D. Harada, and S. J. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proceedings of the Sixteenth International Conference on Machine Learning*, ser. ICML '99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, p. 278–287.

[16] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI'08. AAAI Press, 2008, p. 1433–1438.