Inverse Reinforcement Learning for Dynamics

McKane Andrus

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2019-98
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-98.html

May 26, 2019

Acknowledgement

I'd like to thank Prof. Anca Dragan, first and foremost, for her thoughtful mentorship and unceasing understanding in the face of the many difficulties this project posed. I also want to thank Sid Reddy for the instructive conversations that got this project of the ground and for a good deal of the boilerplate code. I owe a huge thanks to Michael Dennis and Nathaniel Weinman for their immeasurable assistance on the concluding theoretical, infrastructural, and presentational portions of this project. Finally, a massive thanks goes to my partner, Miranda Clemmons, for her unending love and attention, all too often not on terms of her own choosing, but on those dictated by the intensity of my schedule.

Inverse Reinforcement Learning for Dynamics

by

McKane Andrus

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Anca Dragan, Chair
Professor Sergey Levine

Spring 2019

The thesis of McKane Andrus, titled Inverse Reinforcement Learning for Dynamics, is approved:

Chair ____________________________ Date 5/23/2019

____________________________ Date __________

____________________________ Date __________

University of California, Berkeley

Inverse Reinforcement Learning for Dynamics

Abstract

Inverse Reinforcement Learning for Dynamics

by

McKane Andrus

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Anca Dragan, Chair

Inverse Reinforcement Learning (IRL), as standardly defined, entails learning an unknown reward function of a Markov Decision Process (MDP) from demonstrations. As Herman et al. [10] describe, however, learning a reward function from demonstrations is highly reliant on having a correct transition function, which is not always a given. We take interest in this problem from the perspective of Human-Robot Interaction, where we can often observe demonstrations of known tasks but do not have a necessarily correct model of human capability. As such, we specifically consider the case where the goal, or reward, of the demonstrations is known, but the dynamics, or transition function, are not. We refer to this alternate formulation as Inverse Reinforcement Learning for Dynamics (IRLD).

In Chapter 2 we formalize the IRLD problem statement and provide a method that is able to learn unobserved dynamics in environments of limited size and scope. We compare this method to one that estimates both the reward and the dynamics to show the advantages of incorporating knowledge of a goal. In Chapter 3 we propose an alternate, scalable approach to the IRLD problem statement that permits numerous variants of Maximum Likelihood Estimation algorithms using function approximators. Though we were not able to produce a generally applicable method, in chapter 4 we provide a roadmap of the various algorithms we developed and issues that must be addressed for this approach to succeed in future work. Chapter 5 contains a reflection section where our work is positioned in a broader social context.

To Anne Andrus

The most caring and engaging mom a boy could have asked for. From yanking me out of a stifling (and borderline abusive) elementary school to teach me the value of curiosity-driven education to dragging four tired and reluctant kids around the world for a year to show them the breadth of the human condition, the impact of your inimitable love and tutelage is felt daily.

Contents

List of Figures

List of Tables

Acknowledgments

I'd like to thank Prof. Anca Dragan, first and foremost, for her thoughtful mentorship and unceasing understanding in the face of the many difficulties this project posed. I also want to thank Sid Reddy for the instructive conversations that got this project of the ground and for a good deal of the boilerplate code. I owe a huge thanks to Michael Dennis and Nathaniel Weinman for their immeasurable assistance on the concluding theoretical, infrastructural, and presentational portions of this project. A whole-hearted thank you also goes out to those who work in the Interact Lab, they were a source of companionship and solidarity through it all, but especially when this project was at its harshest bend. Finally, a massive thanks goes to my partner, Miranda Clemmons, for her unending love and attention, all too often not on terms of her own choosing, but on those dictated by the intensity of my schedule.

Chapter 1

The Place for Capability Inference in Human Robot Interaction

1.1 Context

As robots and humans come to occupy more of the same environments, seamless interaction becomes an increasingly important goal. For human-to-human interaction, we are able to operate in the same locations in large part because of our own internal models of how those around us interact with the environment. We have a general idea of both what our spatial neighbors are likely to do and what they are capable of doing. These intuitions, however, are highly complex and difficult to mirror in artificial systems. Bestick et al. [4] explore the fault lines of this dearth of intuition in their exploration of the notably difficult yet mundane task of handing another agent an object. In this case, it might be possible to build a general human model for the robot to reason with, but there will be times where the human has a disability or restriction, such as an amputated limb, and the robot will need to learn what those restrictions are. Given observations of the human executing some known task, it should be possible for the robot to identify the difference in behaviour and update its model, as the human's plan will reflect their condition even if it is not visible. We see this work as an early attempt at designing robots for inclusion, actively resisting the defaults-based approaches that tend to be inaccessible or maladapted to individuals of varying disabilities [8].

1.2 Previous Approaches

Up to this point, there have not been many proposed solutions to IRLD. Approaches like those taken by OhnBar, Kitani, and Asakawa [14] generally assume prior knowledge of the dynamics, such that they can update their human model based on a measure of which one provides the best predictive accuracy. Herman et al. [10] outline a method for the "Simultaneous Estimation of Reward and Dynamics" (SERD) in MDPs by maximizing the likelihood of the demonstrations. Crucially, they rely on an exact tabular method that they call "soft Q-Gradient iteration," which can be broken down into maximizing the sum of the log-likelihoods of the actions taken and the log-likelihoods of the transitions observed. As the algorithm requires solving a linear system of equations on the order of the parameters, states, and actions for each gradient step, it becomes intractable for moderately sized state- and action-spaces, not to mention continuous state spaces.

Reddy, Dragan, and Levine [16] tackle a problem similar to IRLD, but instead focus on learning the human's *internal* dynamics model, meaning that they solve for the dynamics model that best describes the observed behavior, not one that both describes the real-world dynamics observations as well as the observed behavior. Unlike the task we would like to solve, they assume access to the true dynamics, which serves as an important reference point for regularizing their learned internal dynamics model. Building off this work, we re-commit ourselves to the assumption that an agent's *internal* dynamics model is the same as the real world's in order to infer *unobserved* components of the real world dynamics. To do this, we adjust the method used by Reddy, Dragan, and Levine [16] to approximately solve the IRLD problem with neural nets or other function approximators.

Chapter 2

IRLD in Finite State and Action MDPs

2.1 Formalizations of IRL

Define a Markov Decision Process (MDP) as a tuple $M = \{S, A, T, \gamma, R\}$, where S denotes the state space, A the action space, T the transition function defined over state, action, and next state tuples (s, a, s'), γ the discount factor, and R the reward function defined over states. Given an MDP M, there exists an optimal policy π^* that maximizes the expected cumulative reward for an agent acting in the MDP. Reinforcement Learning (RL) is generally centered around learning this optimal policy.

Inverse Reinforcement Learning (IRL) [13] is the reverse problem of learning the reward function of an MDP given some observed behaviors of an agent assumed to be operating under π^*. With respect to the MDP formalization, the IRL addresses the case where we receive $M/R + D_{\pi^*}$, or $M = \{S, A, T, \gamma, D_{\pi^*}\}$, where D_{π^*} is a set of demonstration trajectories, and we would like to recover the true rewards R.

An increasingly common approach to solving the IRL problem is to use the Maximum Causal Entropy (MCE) framework [26, 5]. MCE IRL models the demonstrations as being drawn from a policy π_{soft}, defined as:

$$\pi^{\text{soft}}(a|s) = \frac{\exp(\beta Q^{\text{soft}}(s,a))}{\sum_{a' \in A} \exp(\beta Q^{\text{soft}}(s,a'))}. \tag{2.1}$$

In this formulation, β acts as what is referred to as a "rationality coefficient," [7] determining the bias with which the agent selects higher value actions. While we do not assume that this model is sufficient to model human behavior, there is a grounding for this method in psychology, where it is referred to as bounded [15] or noisy [3] rationality, so we accept it for our current purposes. The Qs in this formulation are subject to optimizing the soft Bellman Equations

$$Q^{\text{soft}}(s,a) = R(s,a) + \sum_{s'} T(s,a,s')V^{\text{soft}}(s') \tag{2.2}$$

$$\begin{aligned} V^{\text{soft}}(s) &= \underset{a}{\text{softmax}}(\beta Q^{\text{soft}}(s,a))) \\ &= \sum_a \frac{e^{\beta Q^{\text{soft}}(s,a)}}{\sum_{a' \in A} e^{\beta Q^{\text{soft}}(s,a')}} Q^{\text{soft}}(s,a). \end{aligned} \tag{2.3}$$

For this work we conducted tests using both the weighted log-sum-exp [9]:

$$V^{\text{soft}}_{\text{wlse}}(s) = \frac{\log\left(\sum_{a \in A} e^{\beta Q^{\text{soft}}(s,a)}\right)}{\beta}, \tag{2.4}$$

and the mellowmax [1] variants of the soft value function:

$$V^{\text{soft}}_{\text{mellow}}(s) = \frac{\log\left(\frac{1}{|A|}\sum_{a \in A} e^{\beta Q^{\text{soft}}(s,a)}\right)}{\beta}. \tag{2.5}$$

While the Boltzmann softmax value function given in (2.3) is a more principled definition of the expected value of each state, from [1] we know that it is not a contraction mapping, meaning that there are multiple solutions to the Bellman equations it induces. The two alternate variants, on the other hand, are contraction mappings. The solution to the weighted log-sum-exp Bellman equations is a slight overestimation of the solution to the hard-max Bellman equations, with the degree of overestimation depending on the the magnitude of β. The mellowmax Bellman equations, on the other hand, are guaranteed to be an underestimation of the hard-max solution [1].

With the Bellman equations thus defined, the log likelihood of an observed expert trajectory not including side information (i.e. initial state distributions and transitions) is:

$$L(\tau) = \sum_{(s,a) \in \tau} \log \pi^{\text{soft}}(a|s), \tag{2.6}$$

with causal entropy:

$$H(\pi^{\text{soft}}) = \mathbb{E}_{\tau \sim \pi^{\text{soft}}}(-L(\tau)). \tag{2.7}$$

The solution to the MCE IRL formalization is then the maximization of (2.7) subject to a constraint that the expected feature counts of trajectories from the induced policy are the same as the observed feature counts.

2.2 Formalizing IRLD

Unlike the standard definition of IRL, the problem that we are looking to solve is $M/T + D_{\pi_\theta}$, where the demonstrations are drawn from a policy that had access to the dynamics parameterization θ. Thus, we want to infer the dynamics given the demonstrations and the reward function. Though nearly all past formalizations of IRL require prior knowledge of the dynamics, Herman et al. [10] provide a framework that learns both the rewards and dynamics of an MDP, aptly named Simultaneous Estimation of Rewards and Dynamics (SERD) IRL. For the finite state and action MDP, this formalization is easily adapted to our setting, where we assume knowledge of the reward or goal, but not the dynamics. We provide the formalization of the reduced Estimation of Dynamics (ED) IRL problem below.

We begin with the parameterization of the Q function with respect to the dynamics parameters θ:

$$Q_\theta^{\text{soft}}(s,a) = R(s,a) + \sum_{s'} T_\theta(s,a,s') V_\theta^{\text{soft}}(s'). \tag{2.8}$$

The log likelihood of a demonstration, or set of expert trajectories is then:

$$\begin{aligned} L_\theta(D) &\coloneqq \log P(D|\theta) && (2.9)\\ &= \sum_{\tau \in D} \left[\log P(s_0) + \sum_{(s,a,s') \in \tau} \left[\log \pi_\theta^{\text{soft}}(a|s) + \log T_\theta(s'|s,a) \right] \right] && (2.10) \end{aligned}$$

$$\text{subject to } D \in W_{\text{Observed}}$$

We are then interested in finding the maximum likelihood estimate (MLE) of θ:

$$\theta^* = \operatorname*{argmax}_\theta L_\theta(D) \tag{2.11}$$

Drawing from [10] and [5], we can optimize this objective via a gradient-based method if we analytically define the gradients. Derived in the formulation of Maximum Discounted Causal Entropy (MDCE) IRL [5], the gradient of the log policy, $\nabla_\theta \log \pi_\theta^{\text{soft}}(a|s)$, is built up by the partial derivatives

$$\frac{\partial}{\partial \theta_i} \log \pi_\theta^{\text{soft}}(a|s) = \frac{\partial}{\partial \theta_i} Q_\theta^{\text{soft}}(s,a) - \mathbb{E}_{a' \sim \pi_\theta^{\text{soft}}(a'|s)} \left[\frac{\partial}{\partial \theta_i} Q_\theta^{\text{soft}}(s,a') \right] \tag{2.12}$$

Herman et al. [10] treat the state-action specific Q-value partial derivative differently depending on whether the parameter is a part of the reward or dynamics model, but here we only consider parameters in the dynamics model. The transition-parameter partial derivative is as given:

$$\begin{aligned}\frac{\partial}{\partial\theta_i}Q_\theta^{\text{soft}}(s,a) =&\gamma\sum_{s'\in S}\left[\left(\frac{\partial}{\partial\theta_i}T_\theta(s'|s,a)\right)V_\theta^{\text{soft}}(s')\right] && (2.13)\\ &+\gamma\sum_{s'\in S}\left\{T_\theta(s'|s,a)\mathbb{E}_{a'\sim\pi_\theta^{\text{soft}}(a'|s')}\left[\frac{\partial}{\partial\theta_i}Q_\theta^{\text{soft}}(s',a')\right]\right\} && (2.14)\end{aligned}$$

The gradient of $\log T_\theta(s'|s,a)$ is model specific, and is assumed to be well-defined. Using these evaluations, it is possible to solve $\nabla Q_\theta^{\text{soft}}$ as a system of equations, or iteratively as with Q-learning. This method is known as soft-Q gradient iteration [10].

2.3 Comparing SERD and ED

Seeing as ED is far more well-determined than SERD, we expect it to have a number of advantages. In this section we show that ED does outperform SERD on a number of metrics and in certain environments. This then provides the foundation for chapter 3, where we seek to exploit these determinedness advantages in order to develop a more general dynamics learning method that can handle both continuous and large state-space MDPs assuming that we know the agent's goal.

Model Description

We evaluate our modified Estimation of Dynamics (ED) algorithm on a grid world environment where we can gather our data by simulation and validate against ground truth. We begin with a consciously constructed grid world with known optimal behavior to make it easy to qualitatively test accuracy. We then continue to test on semi-randomly generated grid worlds to quantitatively check the ability for our method to work in generalized environments. To generate these grid worlds, we define a 'clustering factor' to encourage like tiles to be located close to one another. This soft constraint leads to maps that are traversible by the demonstration agent, as bad areas can be avoided. We specify our grid worlds to match previous work [10, 25]. Each grid world has $N \times M$ states organized into a $N \times M$ grid. There are two tile types that can be associated with a state, which we will refer to as *land* and *pits*. There will also be a separate texture map which specifies the reward in each state.

In each state there are 5 available actions $\{up, right, down, left, stay\}$. The effect of each of these actions depends on the tile type. In *pits*, any action you take will result in no movement, remaining in the same state in the next time step. On *land*, all of the actions will act as expected the majority of the time; up will usually go up, down will usually go down. However, for all of the directional actions there is a small probability that the agent will 'slip' and end up in a square in a perpendicular direction to the one in which it intended to go, we will refer to this probability as the 'slip factor'. If an agent tries to go up it will end up one square to the left or one square to the right with probability 'slip factor'. An agent will never

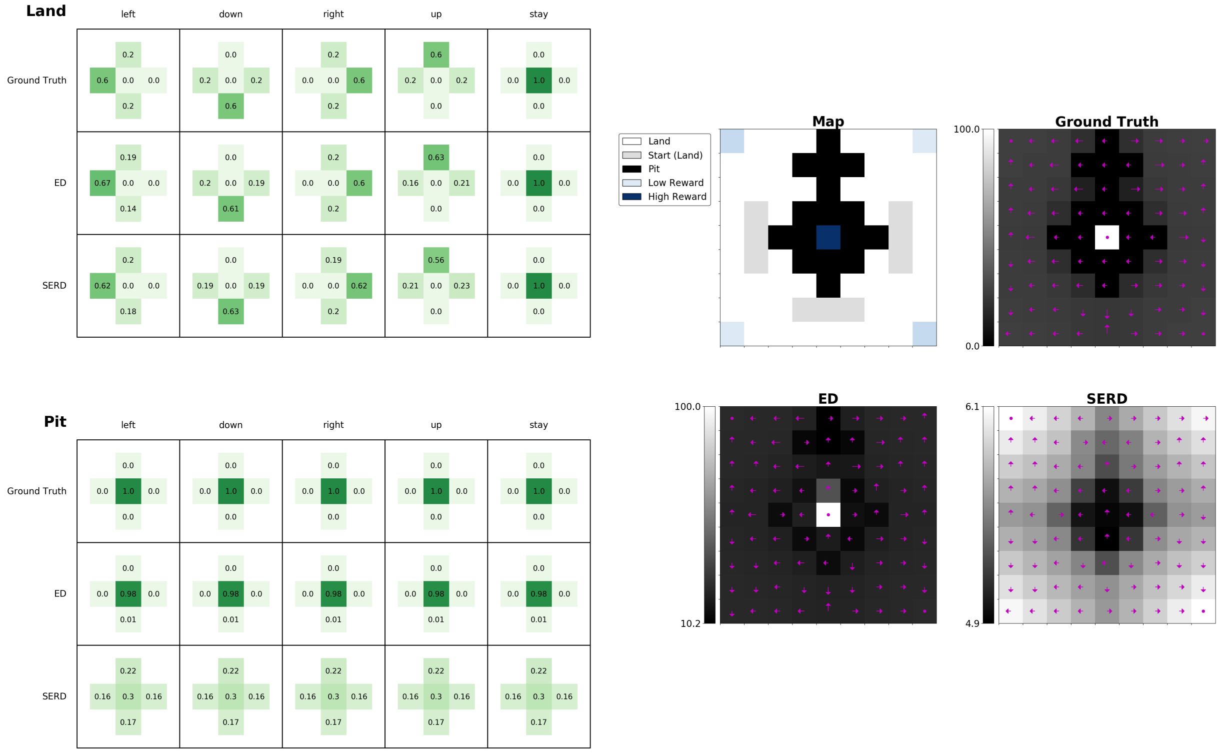

Figure 2.1: Ground Truth and Method Learned Dynamics and Q-Values of a grid world with known optimal behavior. The left diagram displays the learned dynamics for each tile type, action, and method. The right diagram shows the learned value function of each method. The arrows in each tile show that state's most likely action, while the magnitude of the arrow encodes how much more likely that action is over the others.

slip while trying to stay in place. For simplicity we will allow the map to wrap around such that agents that should move up from the top square arrive at the bottom of the map and vice-versa. Similarly, agents that move left from the left most squares will show up in the corresponding rightmost square.

Experimental Results

In the pre-constructed map there is a large reward in the middle, but this reward is surrounded by *pits*. Therefore, all demonstrations will show that the agent avoids the center, instead going for the lower rewards on the corners. From this, an observer who knows that the center square is the agent's true goal should reasonably infer that the agent knows that it cannot make it through the pits surrounding the high reward so it settles for the lesser goal. Figure 2.1 shows that ED can successfully infer these dynamics, while SERD instead finds an explanation that puts extremely low reward on the center square, as we would expect. To be

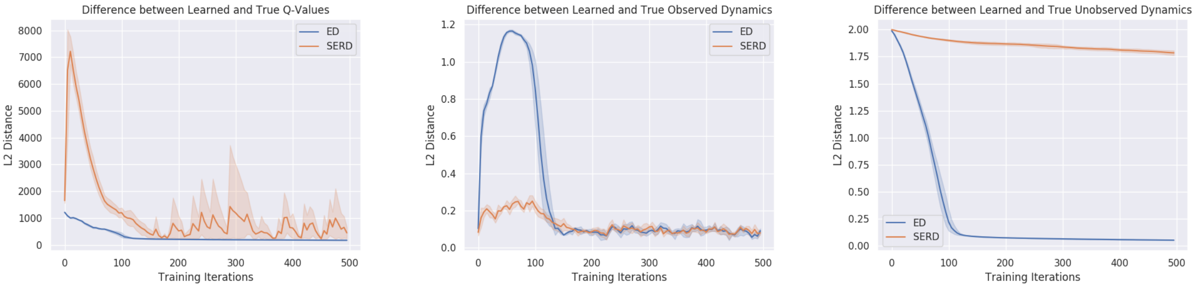

Figure 2.2: A comparison of SERD and ED Q-Value and Dynamics errors of a grid world with known optimal behavior. This is an example of where SERD falls prey to unidentifiability.

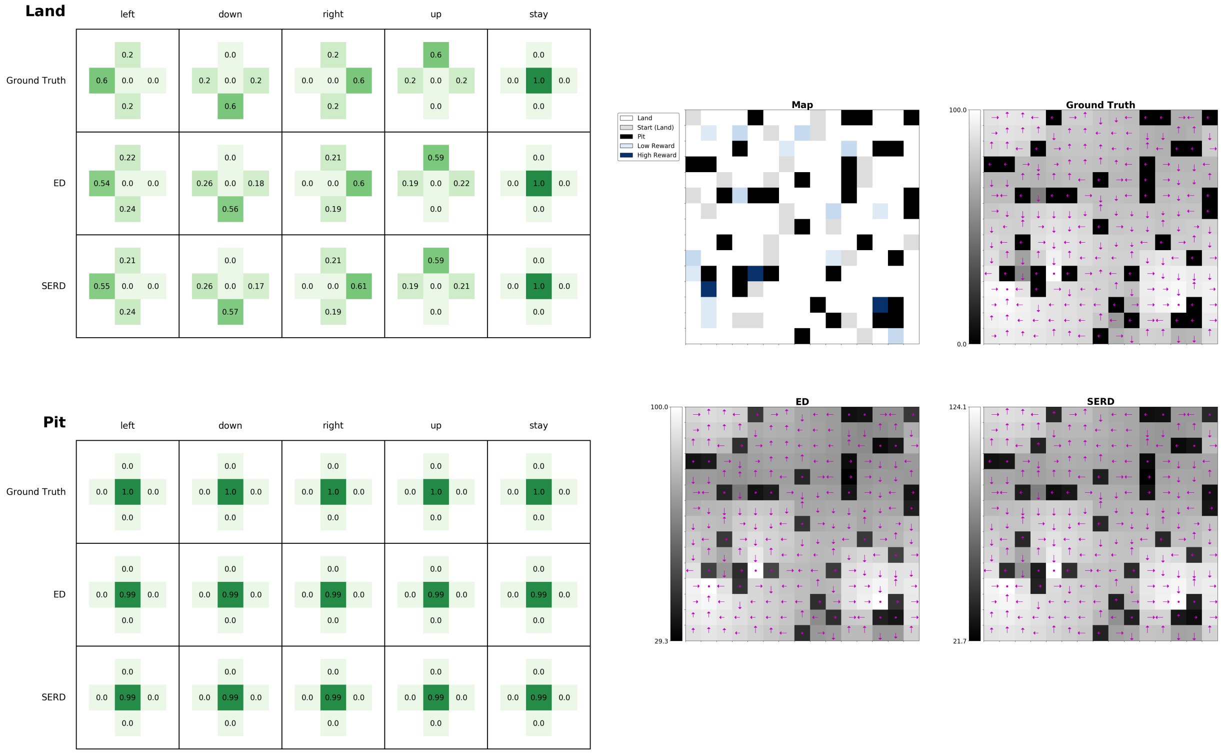

Figure 2.3: Ground Truth and Method Learned Dynamics and Q-Values in a semi-randomly generated grid world.

clear, this is not a failing of SERD, this is just an extreme case of prior reward knowledge being necessary to learn the dynamics.

In Figure 2.3 we see the opposite case. The semi-random maps that we generated trajectories on could all be solved by SERD as well as by ED. Thus, we can assume reward information is not always necessary to learn the true dynamics. At the very least, learning will not be hindered by obtaining accurate reward information. With mis-specified reward information, however, our experiments suggest that a less general dynamics model will be learned. As such, before this method is implemented in non-simulated settings, high confidence in the rewards is required. Realistically, this is only possible in the case of known goals, or where the learner has prior knowledge over the agent's task.

As a brief aside, for the randomly generated grid-worlds seen in Figure 2.3, they are created such that, even with the 'slip factor', the demonstrations of the agents do not result in moving into pits. If this were allowed, we would either need to prune trajectories, biasing the observed dynamics, or cut trajectories short when a *pit* is entered, adding a survivorship bias to the observed transitions.

Chapter 3

IRLD in General MDPs

3.1 A General Approach to IRLD

Transitioning to less restricted MDPs with either very large or continuous state-spaces requires us to abandon the analytical solutions to the soft Q-values and soft Q-gradients that we had in the previous section. We instead model our Q-function with a function approximator as is common in Deep Reinforcement Learning [12, 22, 9, 18, 25], now denoted as $Q_\phi(\cdot, \cdot)$. Now, in order to find a best fit model for our demonstrations, we require two components, a dynamics model that best fits the observed transitions and a policy model that best matches the agents observed actions. Crucially, we assume that there are unobserved parts of the dynamics that influence the likelihood of the observed policy. Depending on the strength of this influence, by optimizing over the likelihood of the demonstrated actions our method should make inferences over the types of unobserved dynamics that could motivate the observed behavior. By ensuring that our policy and dynamics models are consistent with each other, meaning that $Q_\phi(\cdot, \cdot)$ is close to the optimal solution to the MDP $M = \{S, A, T_\theta, \gamma, R\}$, we enable a transfer of information from the observed policy to the unobserved components of our dynamics model. For simplicity we assume the ground truth discount factor γ to be known, but this is generally not a safe assumption outside of simulation, meaning that it must be learned before our method begins or as a variable in our method.

Given our parameterization of the Q function we now require a different formalization of the IRLD objective. Firstly, we consider an updated definition of the log likelihood of the demonstrations:

$$
\begin{aligned}
L_{\theta,\phi}(D) &\coloneqq \log P(D|\theta, \phi) && (3.1)\\
&= \sum_{\tau \in D} \left[\log P(s_0) + \sum_{(s,a,s') \in \tau} \left[\log \pi_\phi(a|s) + \log T_\theta(s'|s,a) \right] \right] && (3.2)
\end{aligned}
$$

$$\text{subject to } D \in W_{\text{Observed}}$$

SERD relies on gradient updates over a fully parameterized Q-function that are intractable for continuous environments. Our method moves towards extending the dynamics learning component of SERD to these types of settings where we make fewer assumptions about the form of the dynamics and policy models, similar to the neural approach used by Wulfmeier et al. [25]. In order to do so, we still need our parameterizations θ and ϕ to be consistent (or nearly consistent) with the soft-Bellman equations, (2.2) and (2.3), or their variants, (2.4) and (2.5). To do this, we first define the soft-Bellman error of a state action pair, $\delta_{\phi,\theta}(s,a)$, to be equal to the following.

$$\delta_{\phi,\theta}(s,a) \coloneqq Q_\phi(s,a) - \left(R(s,a) + \gamma \int_{s'} T_\theta(s,a,s') V_\phi(s') \mathrm{d}s'\right) \tag{3.3}$$
$$\forall s \in S, a \in A$$

As opposed to only considering state-action pairs observed in the demonstrations, we want the Q-values to be consistent with the dynamics over all state action pairs, and as such define δ over all S and A. Equipped with these Bellman residuals, our desired parameters θ and ϕ are the solutions to the following constrained optimization problem:

$$\text{minimize}_{\theta,\phi} \sum_{\tau \in D} \sum_{(s,a,s') \in \tau} -\log T_\theta(s,a,s') - \log \pi_\phi(a|s) \tag{3.4}$$
$$\text{subject to } \delta_{\phi,\theta}(s,a) = 0 \;\; \forall (s,a) \in (W_{\text{Observed}} + W_{\text{Unobserved}})$$

However, we would like our models to be general function approximators, so the objective should be trainable with standard gradient methods, motivating a relaxation of the constraint as in [16]. The Bellman error is incorporated into the optimization objective with a constant penalty term:

$$\text{minimize}_{\theta,\phi} \left(\sum_{\tau \in D} \sum_{(s,a,s') \in \tau} -\log T_\theta(s,a,s') - \log \pi_\phi(a|s)\right) + \rho \left(\int_{s \in S} \sum_{a \in A} \delta_{\phi,\theta}(s,a)\right) \tag{3.5}$$

When optimizing this objective over MDPs with discrete, finite state spaces, we can treat the constraint integral as a sum. When using this method on continuous state MDPs, we can use constraint sampling as in [16]. Finally, for easier reference, we define the following:

$$L_T \coloneqq \sum_{\tau \in D} \sum_{(s,a,s') \in \tau} -\log T_\theta(s,a,s') \tag{3.6}$$

$$L_\pi \coloneqq \sum_{\tau \in D} \sum_{(s,a,s') \in \tau} -\log \pi_\phi(a|s) \tag{3.7}$$

$$\text{BR}_{\phi,\theta} \coloneqq \int_{s \in S} \sum_{a \in A} \delta_{\phi,\theta}(s,a). \tag{3.8}$$

This allows a more concise restatement of our objective:

$$\text{minimize}_{\theta,\phi}\ L_T + L_\pi + \rho \cdot \text{BR}_{\phi,\theta} \tag{3.9}$$

3.2 Implementation Details

We implement this approach through approximating both the Q-function and the Transition function with MLPs parameterized by ϕ and θ respectively. We make the Bellman-error gradient terms more stable by using a target Q-network [22] and use weight norms [17] to increase training speed. In practice we find the weighted log-sum-exp soft-Bellman residual (2.4) has the best performance, so we use it in most cases. Further details of our method are given in the next chapter.

Chapter 4

Of Methods and Fault Lines

Having formulated the problem statement, we move to a number of empirically encountered fault lines that occur when optimizing our given objective and the algorithms we developed in an attempt to surmount them. As we were unable to develop a generally effective algorithm, we instead provide a road-map of our different lines of inquiry and the numerous pitfalls encountered along the way.

A brief warning: Given that we were ultimately unsuccessful, this is a more narrative exposition of methodology than is standard. Most importantly, it was only after testing Algorithm B.2 on a broader suite of environments that we discovered a core flaw that existed in each earlier iteration of Algorithm B. If this flaw had been found earlier, the entire trajectory of the inquiry would have been fundamentally altered. However, as we believe some of the steps that lead to Algorithm B.1 can be applied to other approaches, we initially present them as we understood them at the time and not as we understand them now. This also allows us to traverse our research as a continuous thread. To avoid any confusion on what worked and what did not, the final section of this chapter explicitly covers what should be used in future work.

4.1 Initial Trajectory

Algorithm A.0: Weighted Optimization

Our formulation of IRLD makes reducing the total loss with a well-tuned penalty parameter ρ an intuitive first approach. Ideally, we would be able to find a good default value of ρ that generalizes well to a number of similar environments. After an extensive search through possible values, however, it became clear that such a ρ does not exist. Instead, each value results in a unique local minimum that exhibits the qualities of a Pareto point. At the local minimum there is a careful balance between the action likelihoods, transition likelihoods, and Bellman residuals, where improving one incurs too great a cost on the others.

Furthermore, in Figures 4.1 and 4.2 we can see that when the value of ρ is low, meaning that the Bellman residuals are not strictly enforced, there is more leeway in arbitrarily shifting

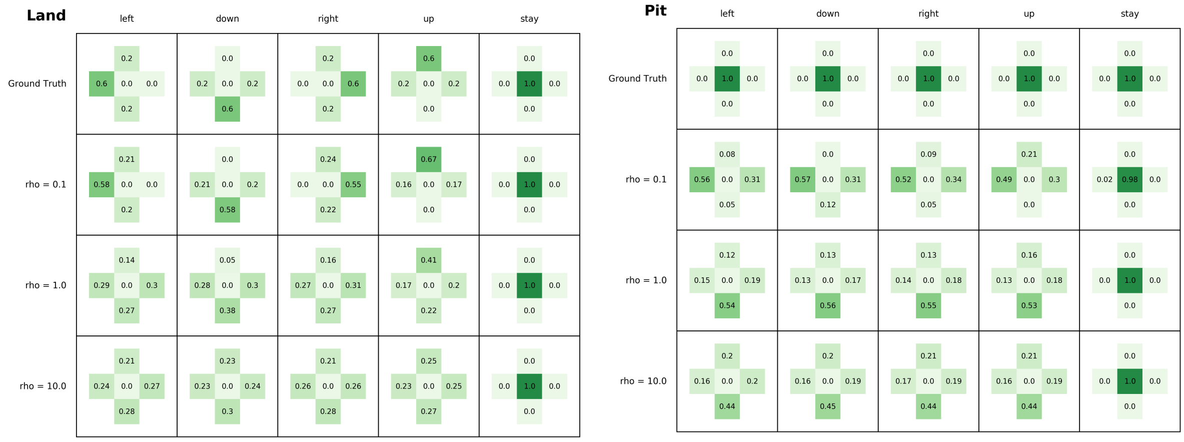

Figure 4.1: A representation of the dynamics learned by Algorithm A.0 for different values of ρ.

the Q-values to improve the action likelihoods, as we would expect. When the value of ρ is increased and the Q-values are more tightly constrained to reality, we see that the observed dynamics are greatly distorted to improve the action likelihood. Figure 4.3 shows how despite the predictive power of the model being massively reduced, the negative log-likelihood of the transitions is not actually increased by much. This calls attention to the first fault line of our IRLD formulation.

Fault Line 1: Likelihood Scales

Unexpected transitions in probabilistic environments are generally along the same order of log-likelihood magnitude as expected transitions. Consider our running example with the 'slip factor': the likely occurrence, moving in the desired direction, has probability $P = 0.6$ ($ll = \log 0.6 \approx -0.22$), where slipping in either orthogonal direction has probability $P = 0.2$ ($ll = \log 0.2 \approx -0.7$). Unexpected actions given a Boltzmann rational actor, on the other hand, can result in log-likelihoods being multiple orders of magnitude larger than transition log-likelihoods. Again using our running example, for an agent with $\beta = 50$ and Q-values tied to the true dynamics, taking the action to move directly into a pit has probability $P \approx 1e{-}13$ ($ll \approx \log 1e{-}13 = -13.0$). Given that our model learns Q-values over mis-specified or unlearned dynamics, we can expect the observed behavior to deviate quite strongly from the expected behavior, resulting in action log-likelihoods multiple orders of magnitude worse than the transition log-likelihoods like those shown above. Thus, while the ground truth models minimize our objective, the directions of steepest gradient descent from most initializations will be those that improve action likelihood with little regard for transition likelihood. Then, once action likelihood and transition likelihood are on the same scale, there will be a tug and pull between them that disallows movement towards the true optimum. So long as Algorithm

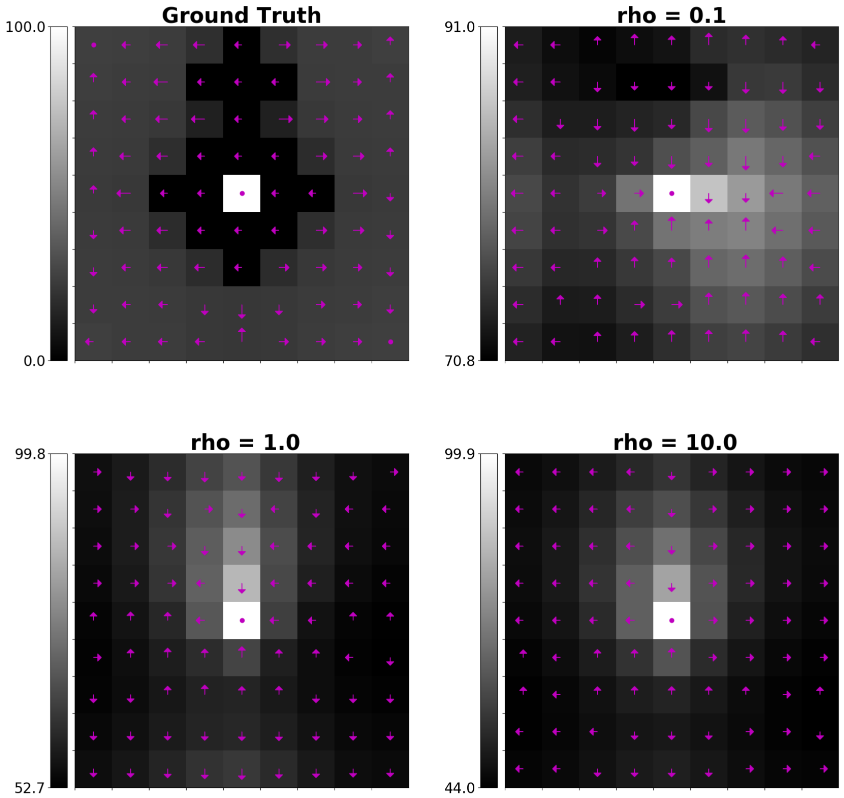

Figure 4.2: A representation of the values learned by Algorithm A.0 for different values of ρ.

A.0 is operating in an environment where this difference in scales exists, we should continue to expect this wildly suboptimal behavior.

Objective Redefinition 1: Weighting the Likelihoods

As motivated by the previous section, we modify our objective, and Algorithm A.0 in turn, by adding individual weights to the transition and action likelihoods.

$$\text{minimize}_{\theta,\phi}\ \lambda_T \cdot L_T + \lambda_\pi \cdot L_\pi + \rho \cdot \text{BR}_{\phi,\theta} \tag{4.1}$$

Algorithm A.1: Separated Likelihood Weighted Optimization

In practice, we find that attempting to balance these new weights is far more art than science. While we would like to weight the transition likelihood sufficiently to put it on the same scale as the action likelihoods, this higher weighting often results in the dynamics model being so tightly optimized over the observations that no changes to it can be made by optimizing the Bellman Residuals. This, however, raises the 'stalemate' point of the Bellman residuals and the action likelihoods. If in turn the weighting on the Bellman residuals is increased, the

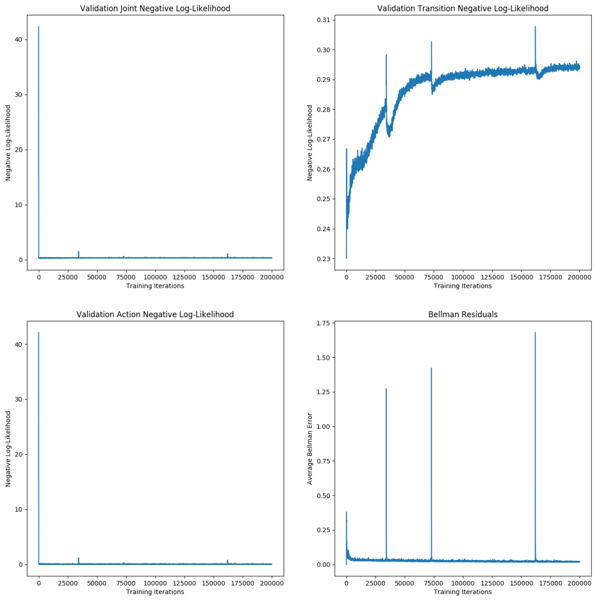

Figure 4.3: Graphs of the values for the negative log-likelihood and Bellman residual terms. The negative log-likelihood for actions hits a peak of over 100× the peak of the negative log-likelihood for transitions.

Q-values are just kept at the optimal solution for the observed dynamics, maintaining a near optimal transition likelihood with Bellman residuals close to zero. In this case the method is effectively just not learning from the observed behavior of the agent. This illustrates that the IRLD objective has a deeper antagonism than just that between the likelihoods.

Fault line 2: Bellman Residual Antagonism

Matching the observed policy will often come at the cost of initially increasing the Bellman residuals, so they then must be reduced by modifying the dynamics and updating the Q values accordingly. This process can be stifled both by the residuals having too strong a pull on the dynamics and by the residuals stalemating with the action likelihoods.

Objective Redefinition 2: Splitting the Bellman Residual

A possible solution to Fault line 2 is to more meticulously manage the antagonism between the IRLD loss components. To do this, we define two new residuals, one parameterized by θ that treats ϕ as given and the other parameterized by ϕ that treats θ as given,

$$\delta_\phi(s,a|\theta) \coloneqq Q_\phi(s,a) - \left(R(s,a) + \gamma \int_{s'} T(s,a,s'|\theta) V_\phi(s')\mathrm{d}s'\right) \tag{4.2}$$

$$\delta_\theta(s,a|\phi) \coloneqq Q(s,a|\phi) - \left(R(s,a) + \gamma \int_{s'} T_\theta(s,a,s') V(s'|\phi)\mathrm{d}s'\right). \tag{4.3}$$

Updating our conciser notation, we now have

$$\mathrm{BR}_\phi \coloneqq \int_{s\in S} \sum_{a\in A} \delta_\phi(s,a|\theta) \tag{4.4}$$

$$\mathrm{BR}_\theta \coloneqq \int_{s\in S} \sum_{a\in A} \delta_\theta(s,a|\phi). \tag{4.5}$$

Finally, the objective is updated to match the new definitions.

$$\mathrm{minimize}_{\theta,\phi}\ \lambda_T \cdot L_T + \lambda_\pi \cdot L_\pi + \rho_T \cdot \mathrm{BR}_\theta + \rho_\pi \cdot \mathrm{BR}_\phi \tag{4.6}$$

This redefinition does not change the θ's or ϕ's that minimize our objective. In fact, the evaluation of this objective is the same as before just with $\rho = \rho_T + \rho_\pi$.

Algorithm A.2: Stop-Gradient Weighted Optimization

This change is implemented by defining two residual losses in our technical method, one with stop gradients over the transitions and the other with stop gradients over evaluations of the Q-function. With this alteration, we can weight δ_θ and δ_ϕ individually. Such granularity allows us to effectively tune each antagonism separately – if the observed dynamics are being quashed, the weight on δ_θ can be lowered, and if the Q-values are not being sufficiently consistent with the dynamics, the weight on δ_ϕ can be raised without conflict. Though we were eventually able to find some success with this method after putting the weights on training schedules, the amount of fine tuning required to get it to work decently on a single instantiation of a single environment led us to seek out a different approach.

Algorithm 1 Weighted Optimization IRLD

```
function IRLD($\tau$, $weights$)
    $\theta \leftarrow \mathrm{MLE}(T_\theta|\tau)$
    $\phi \leftarrow$ Converged optimization of $\mathrm{BR}_\phi$
    $objective \leftarrow weights \cdot [L_T, L_\pi, \mathrm{BR}_\theta, \mathrm{BR}_\phi]^\intercal$
    Optimize $objective$ to convergence
end function
```

Algorithm B.0: Stop-Gradient Coordinate Optimization

Though the previous method showed glimmers of promise, that type of hyperparameter tuning both may not be feasible in more complex cases and is highly dependent on the scale of rewards and Boltzmann coefficient. We instead would like to have a method with minimal tuning and scale-invariance. Towards this end we propose an iterative, coordinate descent training regime for minimizing the IRLD objective. In the previous sections we have seen numerous insidious local minima that seem to make up a Pareto frontier for the weighted method. Intuitively, we would like to navigate around these local minima until we get close to what we know the global optima looks like – very low negative action log-likelihood, a negative transition log-likelihood close to what can be achieved by an MLE estimate over the transitions, and a Bellman Residual that is essentially zero. A coordinate method, if implemented properly, has the benefit of being able to explore beyond that Pareto frontier and push towards this global minimum that we have constructed a priori. Designing the coordinate method, however, is by no means a straightforward task. There are numerous ways our optimizations could be ordered and combined (e.g. train on likelihoods only → train on residuals only → train on both likelihoods and residuals, repeat). While we tested many iterations of coordinate methods, in this paper we present a regime that reflects a deductive reasoning approach to the problem of inferring the dynamics. Our method follows the subsequent steps:

1. Start by ensuring that the dynamics model closely mirrors the observed dynamics in the demonstrations and that the Q-function is initially fine tuned to these dynamics. (optimize L_T and BR_ϕ)
2. Adjusts Qs to match the observed policy, almost assuredly increasing the Bellman Residual. (optimize L_π)
3. With the Bellman Residual high, instead of simply readjusting the Qs, lower the Bellman Residual by manipulating the dynamics. In other words, adjust the dynamics to be most consistent with the Qs inferred from the policy. (optimize BR_θ)
4. Having updated the dynamics, ensure that they are in line with the observed dynamics. Retrain the Q-function to generate values that are consistent with the dynamics model. (optimize L_T and BR_ϕ)
5. Repeat the process by returning to step 2.

Fault line 3: Unintentional Training of Unobserved Dynamics

Looking to Figure 4.4, when the coordinate method reaches step 4 it erases much of what was learned in step 3, even for the unobserved dynamics. Given that in our environment there is no correlation between the two types of dynamics, in theory the unobserved dynamics

Algorithm 2 Coordinate Descent IRLD

```
function IRLD($\tau$)
    $\theta \leftarrow$ MLE($T_\theta|\tau$)
    $\phi \leftarrow$ Converged optimization of BR$_\phi$
    $optimization\_progression \leftarrow [\ [L_\pi],\ [\mathrm{BR}_\theta],\ [L_T, \mathrm{BR}_\phi]\ ]$
    while overall objective not sufficiently converged do
        $objective \leftarrow$ next($optimization\_progression$)
        Optimize $objective$ to convergence
    end while
end function
```

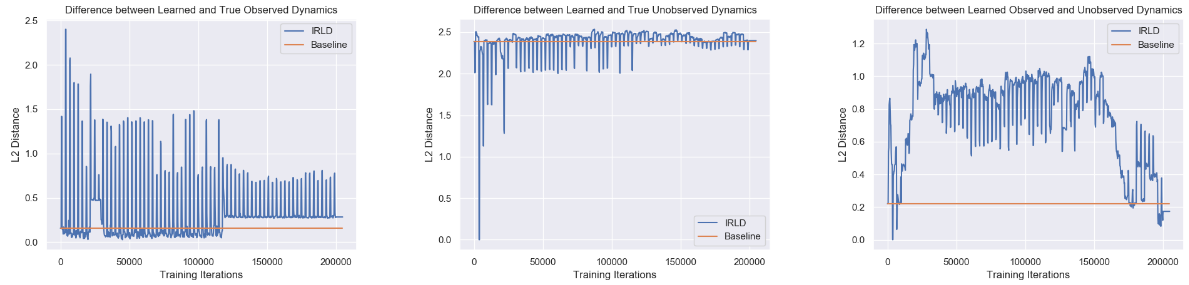

Figure 4.4: The first two graphs from the left show the dynamics errors compared to ground truth in the default grid world for Algorithm B.0. The rightmost graph directly shows the difference between the observed and unobserved dynamics. Steep ticks downwards, meaning the observed and unobserved dynamics are being brought together, occur each time the algorithm starts optimizing L_T and BR$_\phi$.

should not be impacted by optimizing the observed dynamics. The interconnectedness of neural networks, however, makes it such that when trained only on states of type 0 and given no explicit training to distinguish between states of type 0 and 1, the dynamics of type 1 states are pulled towards the dynamics of type 0 states. This effect is not limited to early training where we expect the dynamics to be weakly differentiated; it occurs even after the unobserved dynamics have been uniquely altered.

Objective Redefinition 3: Scale-Invariant Observed Dynamics Enforcement

As one step towards resolving Fault line 3 while also in part addressing Fault lines 1 and 2, we would like to have an optimization term for the dynamics that more strictly enforces its adherence to the observed behavior so that it is less dependent on fine-tuning to the scale of other components of the objective. Inspired by the notion of Trust Regions used elsewhere in Reinforcement Learning [18], on top of optimizing directly over the transition likelihoods we propose a secondary constraint term that encourages the learned dynamics to remain close to the Maximum Likelihood Estimation of θ over the observed transitions, which we denote as

θ', without optimizing so tightly so as to erase the learned unobserved dynamics. We include a confidence measure ϵ that is proportional to the breadth of observation data.

$$\mathrm{D}_{\mathrm{KL}}^{D}(\theta, \theta') := \mathbb{E}_{(s,a,s')\in D}\ \mathrm{D}_{KL}\left(T_\theta(s,a,s') \| T_{\theta'}(s,a,s')\right) \tag{4.7}$$

$$\mathrm{D}_{\mathrm{KL}}^{D}(\theta, \theta') < \epsilon \tag{4.8}$$

Critically, we note that the expected KL-divergence term is only calculated over (s,a,s') triplets in the demonstration space, we say nothing about the distributional distances for the unobserved space. Also, as we have direct access to both probability functions, we calculate the bidirectional KL-divergence to increase stability. We again relax this constraint and adding it to the optimization objective, though this time with the hinge penalty method as it is an inequality constraint.

$$\text{minimize}_{\theta,\phi}\ \lambda_T \cdot L_T + \lambda_\pi \cdot L_\pi + \rho_T \cdot \mathrm{BR}_\theta + \rho_\pi \cdot \mathrm{BR}_\phi + \rho_{\mathrm{KL}} \cdot \max(\mathrm{D}_{\mathrm{KL}}^{D}(\theta, \theta') - \epsilon, 0) \tag{4.9}$$

In this way, by setting a high weight on ρ_{KL} we can effectively form an epsilon-ball of acceptable dynamics exploration around the observed dynamics. We also find that the $\lambda_T \cdot L_T$ term of this admittedly unruly objective is no longer required in practice. By adding this loss term we functionally change the optimal values of θ and ϕ for our objective. As the loss term is tied to data confidence, however, in the limit of data the optimal values are unchanged. Furthermore, the correlation of ϵ to data confidence effectively builds a bubble of L_T losses that should be treated as equivalent.

Algorithm B.1: KL Coordinate Optimization

Given the effects we observed in Figure 4.4, we want to ensure that learned unobserved dynamics are not immediately erased. Now that we have a roughly scale-invariant way of adhering to the observed dynamics, we can have the Bellman residuals adjust the unobserved dynamics without greatly changing the observed dynamics. We modify the coordinate training regime to reflect the following outline:

1. Optimize L_T and BR_ϕ, $\theta' \leftarrow \theta$
2. Optimize L_π
3. Optimize BR_θ and $\max(\mathrm{D}_{\mathrm{KL}}^{D}(\theta, \theta') - \epsilon, 0)$
4. Optimize BR_ϕ
5. Repeat the process by returning to step 2.

In practice we found that putting the KL ϵ on a schedule where it starts a few orders of magnitude above our actual confidence level allows for more early exploration and a steadier guiding towards the global optimum.

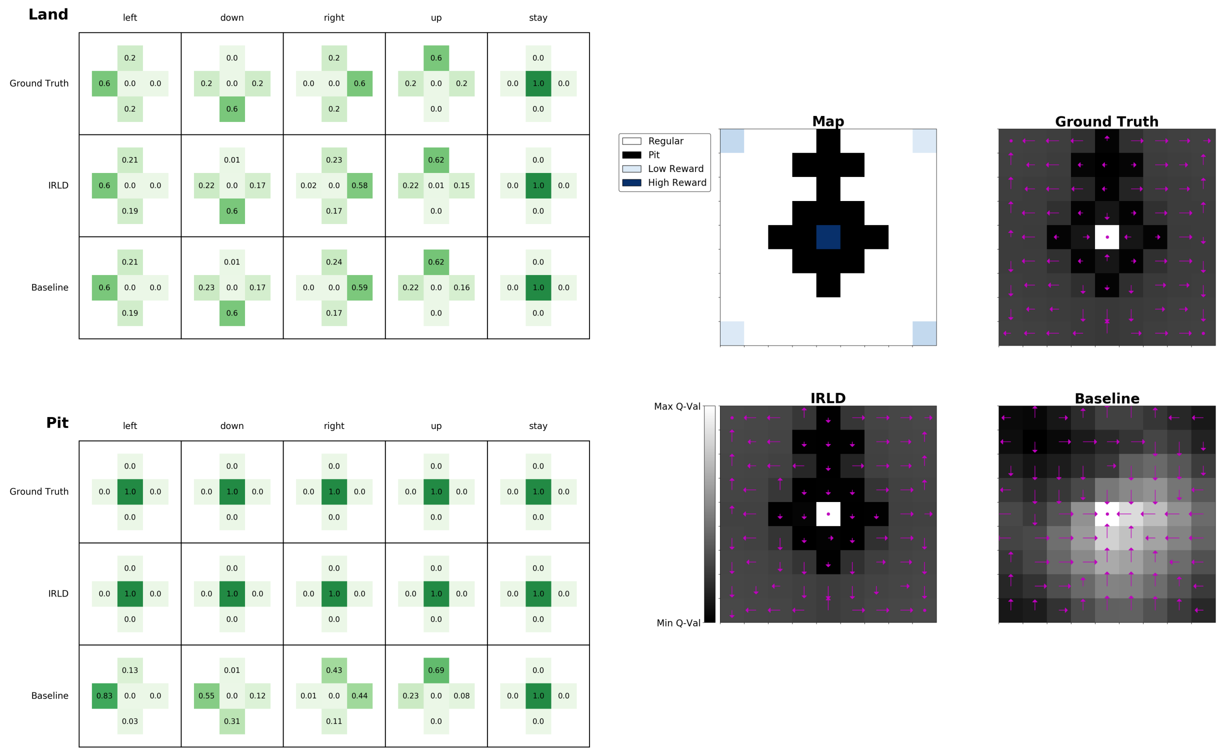

Figure 4.5: Ground Truth and Learned Dynamics and Q-Values of grid world with known optimal behavior

4.2 The Reckoning

In testing the robustness of our algorithm we would often first use our default grid world with various dynamics settings (Figure 2.1) where optimal behavior is well-understood. When it passed those initial tests we would then run it on a set of random grid worlds to get an idea of how well it generalizes. Next on the stack of tests was an environment where no learning should occur. We had not gotten this far until testing Algorithm B.1.

Figure 4.5 shows our method seemingly learning the ground truth dynamics. In Figure 4.6, our method appears to quickly learn the true Q-function and the dynamics. It clearly beats the MLE baseline dynamics and the Q-values trained on those dynamics, evidently learning the unobserved dynamics.

Figure 4.7 seems to show that in these environments, while the MLE baseline method cannot infer the unobserved dynamics, our method consistently can, without suffering loss on the accuracy of the Q-function or on the likelihood of the observed dynamics.

Lastly, we have a map where no learning of the unobserved dynamics should occur, as the unobserved tile-types are too far removed from the start and goal tiles to influence the observed policy. As seen in Figure 4.8, however, our method managed to learn the unobserved

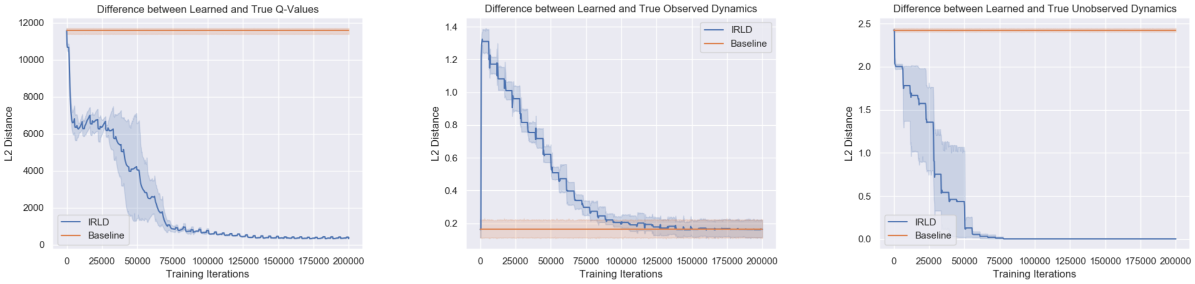

Figure 4.6: Q-Value and Dynamics errors compared to ground truth of grid world with known optimal behavior

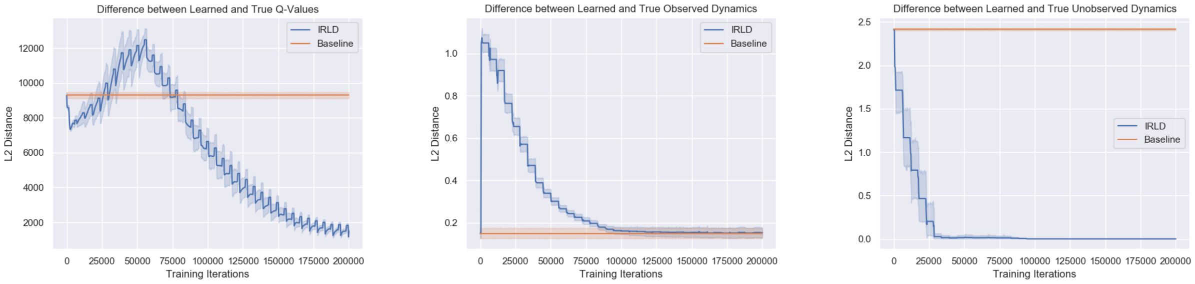

Figure 4.7: Q-Value and Dynamics errors compared to ground truth of randomly generated grid worlds

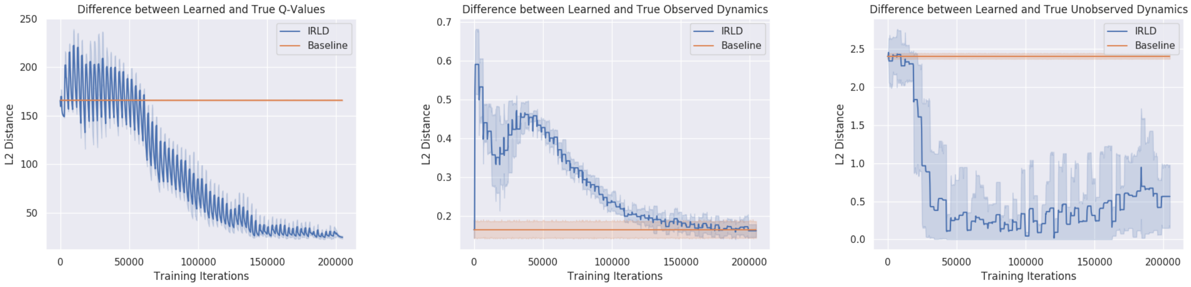

Figure 4.8: Q-Value and Dynamics errors compared to ground truth in a carefully crafted map where no learning of the unobserved dynamics should be possible.

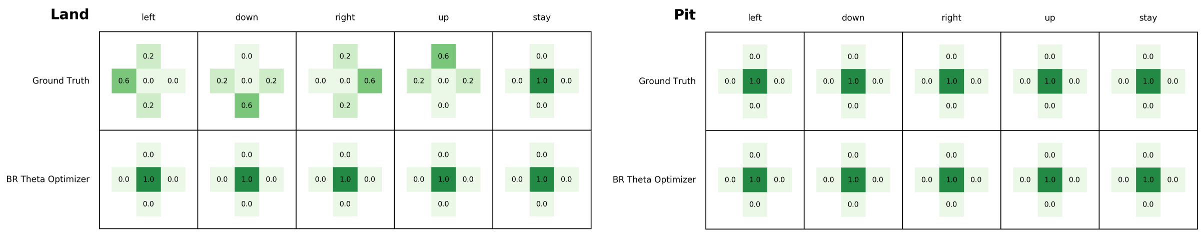

Figure 4.9: The dynamics that result from training only over BR_θ after initializing θ to the MLE estimate and ϕ to the converged Q-values for θ.

dynamics. This counter-intuitive failure of the method illuminated two extremely important fault lines when optimizing the IRLD objective.

Fault line 4: Limited Action Likelihood Propagation

Though this fault line does not explain the failure we observe in Figure 4.8, it was stumbled upon while searching for the source of the failure. In step 2 of the outlines of both Algorithm B.0 and Algorithm B.1, the action likelihoods are optimized over. In step 3, the dynamics are adjusted to better fit the action likelihood influenced Q-values. Quite crucially, however, those Q-values are not consistent (and we did not expect them to be). What this means for fitting a dynamics model to them, however, is that the Q-values in unobserved states are not updated, so there should be no dynamics model that makes the Q-values consistent. What we should instead be optimizing over are the Q-values induced by following the observed policy under the current dynamics model.

Fault line 5: Default Response to Optimizing BR_θ

Given the definition of BR_θ, optimizing it on its own while the value is high is almost guaranteed to push all dynamics to mirror those of the NOOP action if there is one. To see this, consider the case when BR_θ is being optimized immediately following L_π. The value of BR_θ will be high with Q-values that are not consistent. Seeing as Q-values cannot be updated during this stage of optimization, the dynamics will be updated to push $\gamma \int_{s'} T_\theta(s,a,s')V(s'|\phi)\mathrm{d}s'$ as close to the static $Q(s,a|\phi)$ term as possible. Almost always, the θ that accomplishes this is one which puts all the probability mass on any action keeping the agent in the same state, as $V(s|\phi)$ is likely to be closest to $Q(s,a|\phi)$ of all the possible next states. The experiment in Figure 4.9 verifies this intuition. This fault line makes for a difficult roadblock to coordinate methods, as the whole point of optimizing BR_θ on its own (or alongside the transition KL term) is so that the action likelihood weighted Q-values are not immediately overwritten by BR_ϕ, but BR_ϕ is also the only way to propagate the changes made by BR_θ.

Objective Redefinition 4: Distance from Q-values Induced by Observed Policy

As described in Fault line 4, we should be keeping an updated Q-function induced by the observed policy to make meaningful dynamics model updates. We define this induced Q-function as satisfying the following Bellman policy equations:

$$\pi_O \leftarrow \text{MLE}(\pi_\phi|D) \tag{4.10}$$

$$Q^{\pi_O}(s,a) = R(s,a) + \int_{s'} T_\theta(s,a,s')V^{\pi_O}(s') \tag{4.11}$$

$$V^{\pi_O}(s) \coloneqq \sum_a \pi_O(a|s')Q^{\pi_O}(s,a) \tag{4.12}$$

The goal of this addition is to propagate the effects of adhering to the observed policy across the entire MDP, not just in the observed trajectory states. As such, we define a new loss term that can exert this more holistic pressure on the learned Qs:

$$\mathcal{L}_Q = \|Q^{\pi_O}(s,a) - Q_\phi(s,a)\|_2 \tag{4.13}$$

The $Q^{\pi_O}(s,a)$ component is still analytical, however, so we need a parameterization for it. Treating these Bellman policy equations in the same fashion as the original Bellman equations, we have the following definitions that will feed into to new optimization terms:

$$Q^{\pi_O}(s,a) \coloneqq Q_{\hat{\phi}}(s,a) \tag{4.14}$$

$$\delta_{\hat{\phi}}(s,a|\theta) \coloneqq Q_{\hat{\phi}}(s,a) - \left(R(s,a) + \gamma \int_{s'} T(s,a,s'|\theta)V^{\pi_O}_{\hat{\phi}}(s')\mathrm{d}s'\right) \tag{4.15}$$

$$\delta_\theta(s,a|\hat{\phi}) \coloneqq Q(s,a|\hat{\phi}) - \left(R(s,a) + \gamma \int_{s'} T_\theta(s,a,s')V^{\pi_O}(s'|\hat{\phi})\mathrm{d}s'\right). \tag{4.16}$$

Putting this in terms of our conciser notation, we now have

$$\text{BR}^O_{\hat{\phi}} \coloneqq \int_{s\in S} \sum_{a\in A} \delta_{\hat{\phi}}(s,a|\theta) \tag{4.17}$$

$$\text{BR}^O_\theta \coloneqq \int_{s\in S} \sum_{a\in A} \delta_\theta(s,a|\hat{\phi}). \tag{4.18}$$

Future Methods

Though we do not have any noteworthy findings to report for Objective Redefinition 4, it is an area that we are actively exploring. Fault line 5 made clear that coordinate methods with BR_θ at their core are doomed to fail, but we think there may be hope yet for the coordinate method when it makes use of some variant of $\mathcal{L}_Q$. Fault lines 1-4 are all sufficiently addressed given our slew of adjustments to the objective, with the separation of BR_θ and BR_ϕ and the addition of $\mathcal{L}_Q$ seeming to be the most central to overcoming them. Fault line 5, however, calls attention to a deeply fundamental concern that is very difficult to reason about in this high-dimensional problem space. In SERD and ED, learning the unobserved dynamics comes via solving the recursive definition of the Q-value. With this generalized approach we sought to bypass that recursion by directly optimizing over the relaxed objective. Though this type of optimization saw some success in [16], their method learned internal dynamics with access to external dynamics, which they used to regularize the learned internal dynamics. This heavy prior on the dynamics massively reduces the scope of local minima encountered and may also aid in propagating gradients to the unobserved dynamics.

As a final aside, we would be remiss to not mention the countless hours put into implementing, testing, and theorizing about the Mutliple Gradient Descent Algorithm (MGDA) [6, 11]. Inspired by the approach used in [19], where they implemented MGDA for a multi-task computer vision problem, we had hoped that we would be able to transform our objective function riddled with local minima into multiple objectives that could be balanced separately. After extensive experimentation, however, we could not find an implementation that meaningfully handled any combination of the numerous objective terms that we developed.

Chapter 5

Discussion and Conclusions

5.1 Capability Inference Contextualized

As seen in the rise of fields, and conferences in turn, surrounding AI, Machine Learning, ethics, and other social issues, there is increasing social and academic concern for the potential and realized impacts of developments in AI and ML. Though this work has not centered on questions of immediate interest to those fields, we think it is important for us to both do our due diligence in considering the potential impacts of our contribution and to signal a vote of confidence to the work being done in those communities.

Human Modeling for Inclusion or Control?

In the motivation we gave for this work, we discussed how having a mis-specified human model can lead to a number of breakdowns in human-robot interaction. As with most technologies, we can expect that robots developed with an explicit model of a 'normal' human will lead to a disproportionate number of breakdowns for disabled persons [8]. This project is an early attempt at designing robotics for inclusion, and is just one of many exploratory attempts to make human-robot interaction more inclusive in a broader sense (e.g. [21, 23, 20, 2]). Importantly, these efforts are not limited to just robots designed explicitly for rehabilitation and assistance, some of this work is being done for implementation on general social robots, suggesting a trend towards inclusion in robotics research.

We must not, however, forget the political reality of AI research today. The main commercial use of AI and ML is in advertising, recommendation systems, and data analytics. In these contexts, the contexts in which IRLD will most realistically be used, our research helps enable something much more questionable. As Zuboff [27] details, large tech firms' accrual of greater and greater amounts of personal data enables not only behavioral prediction, but behavioral modification that translates into both large profits and reliable social control, however unintentional or well-meaning. Our method could reasonably exacerbate this trend, enabling the deeply objectionable practice of learning about not just what a platform observes a user do, but also what is unobserved.

While it is highly unlikely that our method could be refined to the point where it could learn about user activities outside the platform (Google, Facebook, Amazon, and Apple already have numerous ways of accomplishing this anyways), our main concern is that it could be used to learn a functional representation of the "human dynamics" of a user interacting with a platform. This functional representation would be largely opaque, but it could be used to optimize what is shown to the user and what is not towards some goal. We already see this happening with the Facebook Newsfeed and Youtube Autoplay, where rough human models are optimized over to increase "engagement" or "attention." This is an extremely worrying trend, as the learned behaviors of the systems have little to no human oversight. Just earlier this year we saw that the unfettered optimization of "engagement" by Youtube's recommendation algorithm fostered the growth of pedophilia rings.

All of this being said, we do not believe that IRLD can be classified as one of Winner's "inherently political technologies" [24] – we have already discussed two very different possible use cases, one that is highly centralized with the power to diminish autonomy, and another that is rather decentralized with potential for empowerment. IRLD is instead imbued with the politics of its context, so we ask future practitioners to reflect on the values and stated goals of their task before building on this work.

5.2 Conclusion

In this thesis we presented a formalization of the problem of inferring the dynamics from unobserved parts of the environment. We formulated multiple objectives that could be parameterized with deep neural networks and optimized via gradient decent. We similarly proposed multiple methods to get around the empirical difficulties of optimizing such a complex objective, though none were generally successful. It is our hope that our contributions lay the foundation for future work extending to applications in inferring the dynamics limitations of humans in real-wold environments.

Bibliography

[1] Kavosh Asadi and Michael L Littman. “An alternative softmax operator for reinforcement learning”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 243–252.

[2] Shiri Azenkot, Catherine Feng, and Maya Cakmak. “Enabling building service robots to guide blind people a participatory design approach”. In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE. 2016, pp. 3–10.

[3] Chris L Baker and Joshua B Tenenbaum. “Modeling human plan recognition using Bayesian theory of mind”. In: *Plan, activity, and intent recognition: Theory and practice* (2014), pp. 177–204.

[4] Aaron Bestick et al. “Learning Human Ergonomic Preferences for Handovers”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1–9.

[5] Michael Bloem and Nicholas Bambos. “Infinite time horizon maximum causal entropy inverse reinforcement learning”. In: *53rd IEEE Conference on Decision and Control*. IEEE. 2014, pp. 4911–4916.

[6] Jean-Antoine Désidéri. “Multiple-gradient descent algorithm (MGDA) for multiobjective optimization”. In: *Comptes Rendus Mathematique* 350.5-6 (2012), pp. 313–318.

[7] Jaime F Fisac et al. “Pragmatic-pedagogic value alignment”. In: *arXiv preprint arXiv:1707.06354* (2017).

[8] Alan Foley and Beth A Ferri. “Technology for people, not disabilities: ensuring access and inclusion”. In: *Journal of Research in Special Educational Needs* 12.4 (2012), pp. 192–200.

[9] Tuomas Haarnoja et al. “Reinforcement learning with deep energy-based policies”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1352–1361.

[10] Michael Herman et al. “Inverse reinforcement learning with simultaneous estimation of rewards and dynamics”. In: *Artificial Intelligence and Statistics*. 2016, pp. 102–110.

[11] Quentin Mercier, Fabrice Poirion, and Jean-Antoine Désidéri. “SMGDA: an uncertainty based multi-objective optimization approach. Illustration to an airplane composite material”. In: *Procedia engineering* 199 (2017), pp. 1199–1203.

[12] Volodymyr Mnih et al. “Human-level control through deep reinforcement learning”. In: *Nature* 518.7540 (2015), p. 529.

[13] Andrew Y Ng, Stuart J Russell, et al. “Algorithms for inverse reinforcement learning.” In: *Icml*. Vol. 1. 2000, p. 2.

[14] Eshed OhnBar, Kris Kitani, and Chieko Asakawa. “Personalized dynamics models for adaptive assistive navigation systems”. In: *Conference on Robot Learning*. 2018, pp. 16–39.

[15] Pedro A Ortega and Alan A Stocker. “Human decision-making under limited time”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 100–108.

[16] Sid Reddy, Anca Dragan, and Sergey Levine. “Where do you think you’re going?: Inferring beliefs about dynamics from behavior”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 1454–1465.

[17] Tim Salimans and Durk P Kingma. “Weight normalization: A simple reparameterization to accelerate training of deep neural networks”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 901–909.

[18] John Schulman et al. “Trust region policy optimization”. In: *International Conference on Machine Learning*. 2015, pp. 1889–1897.

[19] Ozan Sener and Vladlen Koltun. “Multi-task learning as multi-objective optimization”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 527–538.

[20] Jainendra Shukla et al. “A case study of robot interaction among individuals with profound and multiple learning disabilities”. In: *International Conference on Social Robotics*. Springer. 2015, pp. 613–622.

[21] Matthias Stöhr, Matthias Schneider, and Christian Henkel. “Adaptive Work Instructions for People with Disabilities in the Context of Human Robot Collaboration”. In: *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*. IEEE. 2018, pp. 301–308.

[22] Hado Van Hasselt, Arthur Guez, and David Silver. “Deep reinforcement learning with double q-learning”. In: *Thirtieth AAAI Conference on Artificial Intelligence*. 2016.

[23] Dinh-Son Vu et al. “Intuitive adaptive orientation control of assistive robots for people living with upper limb disabilities”. In: *2017 International Conference on Rehabilitation Robotics (ICORR)*. IEEE. 2017, pp. 795–800.

[24] Langdon Winner. *The whale and the reactor: A search for limits in an age of high technology*. University of Chicago Press, 2010.

[25] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. “Maximum entropy deep inverse reinforcement learning”. In: *arXiv preprint arXiv:1507.04888* (2015).

[26] Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. “Modeling interaction via the principle of maximum causal entropy”. In: (2010).

[27] Shoshana Zuboff. *The age of surveillance capitalism: the fight for the future at the new frontier of power*. Profile Books, 2019.