Importance Learning: A Data Driven Approach to the Inverse Markov Decision Process with Unknown Transition Probabilities

Connor Thompson[1], Jacob Miller[2], Bo Lin[3], Brian Hartman[1], Timothy C.Y. Chan[3], and Nathan Sandholtz[1]

[1] Brigham Young University, Provo, USA
[2] Sacramento Kings, Sacramento, USA
[3] University of Toronto, Toronto, CAN

Abstract. We introduce an inverse Markov Decision Process (MDP) methodology for when the reward function is known and the transition probabilities are unknown which we call "Importance Learning", specifically designed for data driven contexts. Given a data set consisting of human behavior in a Markov decision process with a known reward function and unknown transition dynamics, we first estimate the transition dynamics based on the data, then solve for the corresponding optimal policy. In the event that observed human behavior deviates from this optimal policy, we propose to explain these deviations by learning a reweighting of the observations such that, when the MDP is estimated and solved based on the reweighted observations, the resulting policy matches the observed human behavior. We interpret these learned weights as a representation of the 'importance' the agent places on that observation when making decisions. Since this is an underdetermined problem, we develop an algorithm that can sample from the solution space, yielding a range of reasonable estimates. We then apply the algorithm to a synthetic gridworld and a real world dataset from the National Football League, displaying the insights we can draw from both.

Keywords: Inverse Markov decision process · Transition probability inference · Data-driven learning · Behavioral modeling

1 INTRODUCTION

The Markov Decision Process (MDP) is a widely used model for sequential decision-making. An MDP is defined by a state space $\mathcal{S}$, an action set $\mathcal{A}$, a transition function $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, describing the probability of moving from state $s \in \mathcal{S}$ to $s' \in \mathcal{S}$ after taking action $a \in \mathcal{A}$, and a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, specifying the immediate utility of that transition. A policy $\pi : S \to \Delta(A)$, prescribes behavior in the environment by mapping each state to a distribution over actions, $\Delta(A)$.

Given p and r, an optimal policy π^* can be determined that maximizes expected long-term reward. We refer to this as the *forward problem.* In the *inverse problem*, a policy $\tilde{\pi}$ representing an agent's observed behavior is given (or can be estimated from data), and the goal is to infer parameters of the MDP that make this policy optimal or approximately optimal. When $\tilde{\pi}$ reflects decisions made under perceived optimality, the inferred MDP can be interpreted as the agent's internal model of the environment that makes their behavior appear rational, even if it diverges from the MDP's true dynamics.

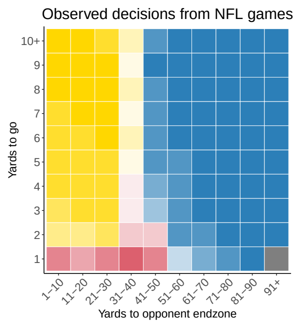
In the existing literature, the inverse problem is almost exclusively formulated by treating the reward function r as the estimand and the transition function p as known. The goal is to estimate an *implied* reward function $\tilde{r}$, such that it makes the observed policy minimally suboptimal conditional on p. In many settings, this is a sensible approach; the reward function guiding the agent is latent and can reasonably be treated as the estimand in the inverse problem. However, in some decision contexts the rewards are explicitly defined by the rules of the environment. In such cases, attributing differences in behavior between observed and optimal actions to alternative reward functions mischaracterizes the problem.

Consider the fourth down decision in the National Football League (NFL). As described in [13], in American football, each team has four "downs" (i.e., chances) to advance the football 10 yards. On fourth down, coaches face a decision: they can go for it, attempting to gain the remainder of the 10 yards and thus a first down, or they can kick the ball either by attempting a field goal (when the team is close enough to the opponent's end zone) or by punting (when the team is farther away). Punting concedes possession of the ball, but it does so by putting the other team in a worse field position, typically deeper in their own half of the field. Attempting a field goal can yield an immediate three points if successful, but a miss gives the opposing team possession of the ball from the location of the kick.

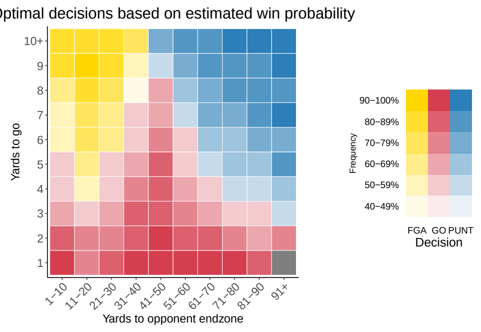
Numerous analyses have shown that coaches have historically chosen systematically suboptimal actions in fourth down situations, as illustrated in Figure 1 [12, 16, 11]. Historically, coaches have behaved far more conservatively than statistical decision models recommend—punting or kicking field goals in situations where going for it yields a higher expected value. To explain this seemingly irrational behavior, existing inverse MDP methods could be used to estimate an implied reward function. However, the resulting inferences could produce misleading conclusions since rewards in football are fixed and known (e.g., six points for a touchdown, three points for a field goal, etc.).

In contrast, other aspects of the decision process may be better suited to explain coach decisions. For example, it is possible that coaches are systematically risk averse [13], or simply are not optimizing only for win probability [12]. Additionally, the transition probabilities between states (e.g., the probability of converting on fourth down) are genuinely unknown, and it is plausible that coaches do not estimate them accurately. In this work, we will treat the agent's implied transition probabilities as the estimand.

The idea of estimating the implied transition function $\tilde{p}$ has received little attention in the literature. To our knowledge, only one prior work, [7], adopts



(a)



(b)

Fig. 1: (a) Most frequent fourth down decisions over the 2014-2022 NFL seasons with respect to yardline (x-axis) and yards to first down (y-axis). Decisions are represented by fill color: yellow denotes field goal attempt (FGA), red denotes go for it (GO), and blue denotes punt (PUNT). The amount of color saturation corresponds to the frequency of the decision—darker colors represent higher frequencies. **(b)** Most frequent fourth down decision prescriptions based on [2] with respect to yardline and yards to first down. Here, color saturation corresponds to the frequency with which the relative majority decision is estimated to be optimal. This figure is inspired by similar figures in [3] and [2] and is identical to that of Figure 1 in [13].

this formulation. They estimate an implied transition function in a model-based manner, analogous to how [6] estimates an implied reward function, constructing a non-convex bilinear program to determine transition probabilities that make a given policy optimal.

Although their method provides for this alternate formulation of the inverse MDP, isn't actually well suited for the fourth down decision problem described above because their central formulation of a nonconvex bilinear program makes it difficult to scale. Additionally, while their method would yield an implied transition function, it would have no interpretation beyond the optimized values.

As far as we are aware, the data-driven problem of learning transition probabilities is absent from the literature. In order to approach it in a data-driven way, it is necessary to first estimate transition probabilities from a dataset. To do this, we will define some parametric form which can model probabilities, such as logistic regression. Then, we introduce weights on each observation in the dataset representing the importance of that observation to the agent's transition probability estimation. The weights will then be used in the model fitting process, shifting away from the empirical estimates while remaining rooted in the data. This weighting mechanism gives a clear interpretation of the solution transition function, linking the results to the observed data.

This subjective choice of a model introduces a large amount of complexity in the process, making direct optimization procedures difficult or impossible. To deal with this, we propose a heuristic method that avoids solving the full inverse MDP, making the optimization much more tractable, even on large datasets.

In this paper, we introduce Importance Learning, a data-driven framework for inferring agent's distorted beliefs about the transition probabilities of an MDP. We infer these distortions by assuming that agents systematically reweight past experiences (operationally, the rows of a data set) when estimating transition probabilities. The objective is to learn a set of weights on past experiences such that, when the MDP is solved using the reweighted data, the resulting policy aligns with the observed behavior. This reweighting formulation provides a practical way to model and quantify cognitive biases such as memory distortion, omission bias, and optimism or pessimism.

Our contributions are as follows:

1) We develop a data-driven framework for inferring the implied transition function in an inverse MDP by systematically reweighting an agent's past experiences as represented by the {s, a, s'}-tuples in a data set of trajectories. This provides an interpretable mechanism for explaining suboptimal decision-making within the MDP framework.
2) To handle the complexity of optimization with our framework, we propose a tractable inference procedure that avoids solving the full inverse MDP or inverse optimization problem, providing a computationally efficient and flexible alternative to existing approaches.
3) We demonstrate the method on both a synthetic MDP and a large dataset of fourth-down decisions from the NFL, showing that the inferred weights yield plausible and interpretable explanations of suboptimal behavioral patterns.
4) We are able to provide interval estimates, rather than just point estimates, showing a reasonable range of solutions.

The remainder of the paper is organized as follows. Section 2 defines the forward problem, Section 3 defines the inverse problem, Section 4 presents the proposed solution framework for the data-driven inverse problem. Section 5 applies the method to a modified version of a gridworld MDP example, and Section 6 demonstrates the approach using data from the NFL fourth-down problem, Section 7 provides a discussion.

2 THE FORWARD PROBLEM

To contextualize the inverse MDP problem we introduce in this paper, we first distinguish between two formulations of the forward MDP problem: a model-based version and a data-based version.

2.1 Model-Based Forward Problem

The *model-based* problem assumes the transition and reward functions are known and solves for an optimal policy under these primitives. Given transition function p, reward function r, and policy π, the *value function* at state s is the expected long-term (possibly discounted) reward beginning in state s and following π thereafter:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right] \\ &= \sum_{a\in\mathcal{A}} \pi(a \mid s) \sum_{s'\in\mathcal{S}} p(s' \mid s, a)\left[r(s,a,s') + \gamma v_\pi(s')\right], \end{aligned} \tag{1}$$

where $\gamma > 0$ is a discount factor.

Given this definition, an optimal policy can be found by solving the following linear program:

$$\begin{aligned} \min_{v} \quad & \sum_{s\in\mathcal{S}} v(s) \\ \text{s.t.} \quad & v(s) \geq \sum_{s'\in\mathcal{S}} p(s' \mid s, a)\left[r(s,a,s') + \gamma v(s')\right], \\ & \forall s \in \mathcal{S},\ a \in \mathcal{A}. \end{aligned} \tag{2}$$

The solution to (2) is the optimal value function, from which an optimal policy π^* satisfying $v_{\pi^*}(s) \geq v_\pi(s)$ for all $s \in \mathcal{S},\ \pi \in \varPi$ can be derived. This is the classical formulation of the forward problem typically associated with solving an MDP [10].

2.2 Data-Driven Forward Problem

The *data-driven* formulation adds a significant layer of complexity to the forward problem. In this case, the reward function is assumed to be known but the transition function is not. Instead, we observe trajectories from a decision maker's interactions with the environment from which the transition function must be estimated. Let $\mathcal{D}$ denote this observed dataset, which contains n tuples (s_i, a_i, s_i') generated from an MDP with known reward function r.

The first step in the data-driven forward problem is to estimate the transition function, which, for generality, we assume follows a parametric form $p(s' \mid s, a; \boldsymbol{\theta})$, $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ This parametric form may be as specific as having a parameter for each transition (e.g., the empirical probabilities), or could be a machine learning model if there is not enough data to estimate the empirical probabilities. The transition parameters are estimated by solving

$$\hat{\boldsymbol{\theta}} = \operatorname*{arg\,min}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \sum_{i=1}^{n} \mathbf{w}_i\, \ell[s_i',\, p(s_i' \mid s_i, a_i; \boldsymbol{\theta})], \tag{3}$$

where $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_n)$ is a vector of nonnegative weights and $\ell(\cdot)$ is an appropriate loss function (e.g., cross-entropy for discrete next-state outcomes).

Although the inclusion of the weight vector $\mathbf{w}$ may seem superfluous, we retain it here for generality. These weights will take on greater meaning in the inverse problem, where each $\mathbf{w}_i$ represents the relative importance the decision maker assigns to their ith experience when forming beliefs about the transition dynamics.

After estimating the transition function $\hat{p} = p(\cdot \mid \hat{\boldsymbol{\theta}})$, the forward problem reduces to solving the linear program in (2), substituting $\hat{p}$ for p. The complete data-driven forward problem can be expressed as the joint optimization:

$$\begin{aligned}
\min_{v,\,\hat{\boldsymbol{\theta}}} \quad & \sum_{s\in\mathcal{S}} v(s) \\
\text{s.t.} \quad & \hat{\boldsymbol{\theta}} \in \operatorname*{arg\,min}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \sum_{i=1}^{n} \mathbf{w}_i \, \ell[s_i',\, p(s_i' \mid s_i, a_i; \boldsymbol{\theta})]\,, \\
& v(s) \geq \sum_{s'\in\mathcal{S}} p(s' \mid s, a; \hat{\boldsymbol{\theta}}) \left[r(s,a,s') + \gamma v(s')\right], \\
& \qquad \forall s \in \mathcal{S},\ a \in \mathcal{A},
\end{aligned}$$

where all terms are as previously defined.

Before continuing to the inverse problem formulation, we make a few comments on the parameterization of $\hat{p}$. At its most complex, $p(s' \mid s, a; \boldsymbol{\theta})$ is completely unstructured, meaning that a unique, non-reducible parameter $\theta_{s',s,a}$ governs each (s', s, a) triple. Learning such a transition model requires observations for all possible transitions in the MDP (i.e., those with $\theta_{s',s,a} > 0$) and prior knowledge of which transitions are impossible (i.e., $\theta_{s',s,a} = 0$). In this fully parameterized case, the dimensionality of $\boldsymbol{\theta}$ is $|\mathcal{S}| \times |\mathcal{S}| \times |\mathcal{A}|$. We illustrate such a case in our synthetic gridworld example (Section 5). This formulation can be severely underdetermined when data are sparse relative to the complexity of the state–action space.

In practice, simpler parameterizations are often necessary. A common approach is to restrict $p(s' \mid s, a; \boldsymbol{\theta})$ to a lower-dimensional model (e.g., logistic or multinomial regression), yielding a parameter vector of substantially smaller dimension. We illustrate such a structured specification in our application to the NFL fourth-down decision problem (Section 6).

3 Inverse Problem Formulation

3.1 Model-Based Version

Inverse optimization in Markov decision processes has traditionally focused on recovering the implied reward function $\tilde{r}$ that makes an observed policy optimal, assuming the transition function p is known. Given a policy π, the goal is to find a reward function $\tilde{r}$ such that π satisfies the Bellman optimality conditions under $\tilde{r}$. As shown by [1] and applied to MDPs by [6], this problem can be formulated as a finite-dimensional linear program by enforcing complementary-slackness conditions between the primal and dual forms of the forward MDP.

More recently, [7] considered the problem of inferring transition probabilities rather than rewards. In their formulation, the reward function $r(s, a, s')$

is assumed known and the objective is to determine fully unstructured transition probabilities that make a given policy optimal. Conceptually, they seek $\tilde{\boldsymbol{\theta}} := \{\tilde{p}(s'|s,a) : s, s' \in \mathcal{S},\ a \in \mathcal{A}\}$ under which an observed policy $\tilde{\pi}$ satisfies the Bellman optimality conditions:

$$\tilde{\boldsymbol{\theta}} \ \in\ \arg\min_{\boldsymbol{\theta}} \left\{ \sum_{s\in\mathcal{S}} v_{\tilde{\pi}}(s) \ :\ v_{\tilde{\pi}}(s) \geq \sum_{s'\in\mathcal{S}} \theta_{s',s,a}\big[r(s,a,s') + \gamma\, v_{\tilde{\pi}}(s')\big], \ \forall s \in \mathcal{S},\ a \in \mathcal{A} \right\}. \tag{4}$$

The precise linear-programming formulation of (4) is a nonconvex bilinear program and can be found in equation (12) of [7].

In empirical settings, the target policy $\tilde{\pi}$ may not be exactly specified *a priori*, but rather reflected through observed trajectories. We next consider this case in the data-driven version of the inverse problem.

3.2 Data Driven Version

In our formulation of the data-driven inverse problem, we assume the same inputs as in the forward problem defined in Section 2.2: a dataset $\mathcal{D} = \{(s_i, a_i, s'_i)\}_{i=1}^n$ of observed transitions and a known reward function $r(s,a,s')$. As in Section 2.2, we assume the transition model follows a parameterized form $p(s'|s,a;\boldsymbol{\theta})$.

In the forward problem, $\boldsymbol{\theta}$ is estimated as in (3), presumably with all $\mathbf{w}_i = 1$, then $\hat{\boldsymbol{\theta}}$ is plugged in to (2) to determine the corresponding optimal policy. In the inverse problem, we assume the observed actions represent the decisions that the decision-maker *believed* to be optimal according to their model of environment as represented by $\tilde{\boldsymbol{\theta}}$—though not necessarily ones that were truly optimal under the environment's dynamics. We are interested in estimating $\tilde{\boldsymbol{\theta}}$ such that it makes their actions in $\mathcal{D}$ approximately optimal.

To operationalize this, we treat the weights $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_N)$ as *unknown*, representing the relative importance that the agent implicitly assigns to each past experience when forming beliefs about the transition dynamics. Let $\pi_{\mathbf{w}}$ denote the policy that is optimal under the transition model $p_{\mathbf{w}}(s'|s,a) = p(s'|s,a;\boldsymbol{\theta}_{\mathbf{w}})$, that is,

$$\pi_{\mathbf{w}}(s) := \arg\max_{a\in\mathcal{A}} \sum_{s'\in\mathcal{S}} p_{\mathbf{w}}(s'|s,a)\big[r(s,a,s') + \gamma\, v_{\pi_{\mathbf{w}}}(s')\big].$$

We seek a weight vector such that an observed policy $\tilde{\pi}$ is optimal. Specifically, we solve

$$\min_{\mathbf{w}\in\mathbb{R}^{|\mathbf{w}|}_{\geq 0}} \ell(\tilde{\pi}, \pi_{\mathbf{w}}) \tag{5}$$

where ℓ is an appropriate loss function which could depend on the problem domain.

In the case where we do not have direct access to $\tilde{\pi}$, we must estimate it. Let $\mathbf{a} = (a_1, \ldots, a_N)$ denote the sequence of observed actions and $\mathbf{s} = (s_1, \ldots, s_N)$ denote the sequence of observed states. We then define

$$\tilde{\pi}(a \mid s) = g(\mathbf{s}, \mathbf{a})(a \mid s), \tag{6}$$

where $\tilde{\pi} : \mathcal{S} \to \Delta(\mathcal{A})$ is the estimated policy mapping each state $s \in \mathcal{S}$ to a distribution over actions $\mathcal{A}$, and g is a model that infers this policy from the observed data.

This formulation can be viewed as a data-driven analogue of the model-based inverse problem in Section 3.1. While [7] solve directly for transition probabilities $\tilde{p}$ satisfying the Bellman equations, our approach infers $\tilde{p}$ indirectly through the weights $\mathbf{w}$ that govern how the transition model is estimated from data. This reweighting framework preserves the interpretability of the inverse MDP while avoiding the need to solve a nonconvex bilinear program.

3.3 Non-Identifiability of Transition Probabilities

While [7] solves the problem by directly optimizing for a probability function that also satisfies a set of constraints, leading to a unique solution, we will not approach the problem in this manner. Although it is possible to perform this procedure with weights when solving Equation (5), there are some large problems with this approach that must be addressed. In general, there is not only one set of probability functions that will make a given policy optimal. In fact, the set of solution probability functions can be infinite. Thus, any point estimate may fail to reflect the agent's true transition model and no statistical guarantees can be made without additional assumptions.

Proposition 1. *Let R and π be fixed. Then there may exist multiple distinct transition models p such that π is optimal under (p, R); that is,*

$$v^p_\pi(s) \geq v^p_{\pi'}(s), \quad \forall \pi' \in \Pi, \ \forall s \in \mathcal{S}.$$

In general, the transition model p is not identifiable from (R, π).

Proof. In Appendix A

Because the transition probabilities may be non-identifiable, we will avoid using any direct optimization procedures to estimate them. Adding additional constraints to an optimization procedure may yield a unique solution, but there is little reason to assume any constraint gives a better estimate of the agent's believed probabilities than any other. Instead, we will develop a procedure that draws samples from the solution set, providing a better idea of the inherent uncertainty of the problem. This is similar in concept to using posterior samples to perform inference and quantify uncertainty in a Bayesian analysis.

Under the assumption that the decision-maker's behavior reflected in $\tilde{\pi}$ really represents them believing they are acting optimally, the solution to the inverse problem—the estimated MDP environment—can represent the decision maker's perceived or believed reality. Human decisions, as reflected in $\tilde{\pi}$, are shaped by experience, limited information, and systematic cognitive biases. Consequently, solving the corresponding inverse problem does not necessarily reveal the true model parameters underlying the decision process, but rather those that the decision maker perceives—the internal model of the world that makes their behavior appear rational from their own point of view.

4 Iterative Reweighting

In order to solve the inverse problem, we use Algorithm 1, which we call Iterative Reweighting. This algorithm works by iteratively updating the observation weights $\mathbf{w} \in \mathbb{R}_+^N$ for a preset number of iterations T, although another stopping criterion may also be used. At each iteration, a proposal function $u : \mathbb{R}_+^N \to \mathbb{R}_+^N$ generates candidate weights. The forward problem is then solved using these proposed weights, and the proposal is accepted only if it decreases the loss between the observed policy $\tilde{\pi}$ and the optimal policy under the weights of that iteration $\pi_{\mathbf{w}^{(0)}}$. Once the algorithm has been run, as many candidate solutions may be retained as desired.

Algorithm 1 Iterative Reweighting

```
1: $\mathbf{w}^{(0)} \leftarrow \mathbf{1}_N$
2: $\tilde{\boldsymbol{\theta}}^{(0)} \leftarrow \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{N}\sum_{t=0}^{N} \mathbf{w}_t^{(0)} \ell[s_t', p(s_t'|a_t, s_t; \boldsymbol{\theta})]$
3: $l^{(0)} \leftarrow \ell(\tilde{\pi}, \pi_{\mathbf{w}^{(0)}})$
4: for $i = 0$ to $T-1$ do
5:     $\mathbf{w}^* \leftarrow u(\mathbf{w}^{(i)})$
6:     $\tilde{\boldsymbol{\theta}}^* \leftarrow \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{N}\sum_{t=0}^{N} \mathbf{w}_t^* \ell[s_t', p(s_t'|a_t, s_t; \boldsymbol{\theta})]$
7:     $l^* \leftarrow \ell(\tilde{\pi}, \pi_{\mathbf{w}^{(0)}})$
8:     if $l^* < l^{(i)}$ then
9:         $\mathbf{w}^{(i+1)} \leftarrow \mathbf{w}^*$
10:        $l^{(i+1)} \leftarrow l^*$
11:    else
12:        $\mathbf{w}^{(i+1)} \leftarrow \mathbf{w}^{(i)}$
13:        $l^{(i+1)} \leftarrow l^{(i)}$
14:    end if
15:    $i \leftarrow i + 1$
16: end for
```

In our use of Algorithm 1, we will use the bootstrap resampling method as the update function. This is equivalent to sampling the vector $\mathbf{w}_{i+1}$ from a multinomial distribution with $\mathbf{p}_t = \mathbf{w}_t/N$, which also limits the solution to non-negative integer weights. While extensions are possible with the use of other proposal functions, we find that the bootstrap proposal works well in practice.

We acknowledge that the greedy nature of this algorithm has inherent exploration drawbacks. To deal with this, often multiple runs of the algorithm will be performed, then solutions will be compiled from all the runs to assess the solution space. This is similar in concept to running multiple chains with an MCMC sampler. We will explore how well this procedure quantifies uncertainty in the estimates in 5.

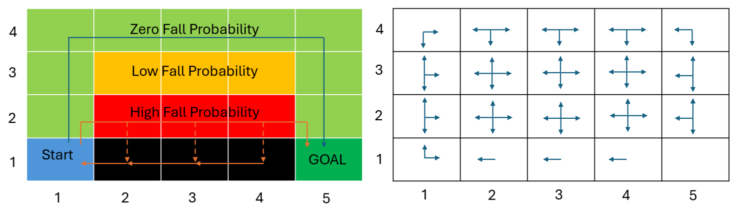


Fig. 2: Crumbly Cliff Walk Environment (Left) and Action Space (Right). In the environment, green cells have no probability of falling. Orange cells have probability p^L of falling, and red cells have probability p^H. Black cells represent the bottom of the cliff. An agent's goal is to get from the start cell in (1,1) to the end cell in (1,5). Two possible policies are shown: the safe policy, going along the top row, and the risky policy, going along the cliff edge for a shorter path, but risking falling.

5 CRUMBLY CLIFF WALK

To illustrate our method we have created a simple MDP which we call a 'Crumbly Cliff Walk', as it is a slight variation of the popular Cliff Walk MDP as found in [15]. We assume a state space consisting of an $a \times b$ gridworld, with a cliff in cells (2,1), (3,1), and (4,1). Every time step comes with a reward of -1, incentivizing finding the quickest path. The 'crumbly' part comes in through setting some probability p^H or p^L of transitioning to the cliff states when in row 1 or row 2 respectively, regardless of the action taken. This adds some amount of risk to the agent taking a path too close to the cliff. Falling off the cliff entails moving directly to the corresponding cliff cell in the same column, which then forces the agent to take actions to move back to the starting square. Figure 2 illustrates this example.

In this MDP, the agent's goal is to efficiently (i.e., minimizing the number of steps taken) get from the start cell to the end cell.

5.1 Forward Problem

First, we must define the forward problem. In this MDP the agent's goal is to efficiently (i.e., minimizing the number of steps taken) get from the start cell to the goal cell. We will assume that the agent has already performed some sort of *exploration* phase, recording its experiences in some dataset D with where D_i contains the tuple (s_i, a_i, s'_i). The rewards are not necessary to record, since we assume the agent has access to the reward function $R(s, a, s')$.

In order to start the *exploitation* process, the agent must use the information captured in D to estimate the optimal policy. First, the agent must estimate $\hat{p}^H_{\mathbf{w}}$ and $\hat{p}^L_{\mathbf{w}}$. We will assume they do this with the form

$$\hat{p}^H_{\mathbf{w}} = \frac{1}{\sum_{i \in D_H} \mathbf{w}_i} \sum_{i \in D_H} \mathbf{w}_i \mathbf{I}(s'_i \in S_C), \tag{7}$$

where D_H is the subset of D where the state s is in the state space with a high probability of falling, $S_H = \{(2,2),(3,2),(4,2)\}$. S_C is the subset of the state space representing cliff cells. Notice that this is a special case of Equation (3), where $\boldsymbol{\theta}$ is simply the proportion of times a specific (s,a,s') tuple is observed.

Once the agent similarly estimates $\hat{p}^L_{\mathbf{w}}$, these transition probabilities can be combined as $\hat{p}_{\mathbf{w}}$ used to find the optimal policy under the weighting $\pi_{\mathbf{w}}$.

5.2 Inverse Problem

Now, suppose we are given an agent's experiences D generated from the Crumbly Cliff Walk environment. From D, we can estimate $\hat{p}^H_{\mathbf{w}}$ and $\hat{p}^L_{\mathbf{w}}$ with weights of 1, then compare $\pi_{\mathbf{w}}$ to the agent's observed policy, $\tilde{\pi}$.

We assume that the policy $\tilde{\pi}$ comes from an agent with experience in the MDP, and who is trying to optimize their long-term reward. However, in the case that $\tilde{\pi} \neq \pi_{\mathbf{w}}$, we must suppose that the agent believes in some set of transition probabilities $(\tilde{p}^H, \tilde{p}^L) \neq (\hat{p}^H, \hat{p}^L)$ which make $\tilde{\pi}$ optimal. Then, the goal is to find a vector of weights $\mathbf{w}$ such that $\tilde{\pi} \neq \pi_{\mathbf{w}}$.

In order to solve the inverse problem, it is necessary to define a loss function to compare the estimated and target policies. We will define the loss between $\tilde{\pi}$ and the optimal policy based on the weighted transition probabilities $\pi_{\mathbf{w}}$ as

$$\ell\left(\tilde{\pi}, \pi_{\mathbf{w}}\right) = \sum_{s \in S} \left|\hat{v}_{\hat{p}_{\mathbf{w}},\pi_{\mathbf{w}}}(s) - \hat{v}_{\hat{p}_{\mathbf{w}},\tilde{\pi}}(s)\right|, \tag{8}$$

where $\hat{v}_{\hat{p}_{\mathbf{w}},\pi}(s)$ is the estimated value of the state s with respect to a given policy π, the transition dynamics $\hat{p}_{\mathbf{w}}$, and the rewards r which are assumed to be known to the agent.

5.3 Solving the Crumbly Cliff Walk

We will explore three distinct policies in this MDP. The 'risky' policy is the one that takes the path along the 2nd row, next to the cliff. The 'safe' policy moves up and travles along the 4th row, as far from the cliff as possible. Finally, the 'medium' policy takes the path along the 3rd row, between the other two.

In this simulation, we set the true values of p_H and p_L to 0.2 and 0.15 respectively. With the reward for each timestep being -1, these dynamics make the safe policy optimal. Next, to collect D we simulate 10 random walks of the gridworld. Statistically, the estimated dynamics from D will be very close to the true dynamics. We then set the target policy $\tilde{\pi}$ to the risky policy, and pass $\tilde{\pi}$ and D into Algorithm 1.

From the results in 15 we can see the distribution of the transition probability estimates over several runs of the bootstrap algorithm. The blue lines show the true values used in the data generating process, and the red lines show pre-selected values for $\tilde{p}^H$ and $\tilde{p}^L$, .1 and .05 respectively, which make $\tilde{\pi}$ optimal. We also performed a grid search to find the true solution range for each probability,

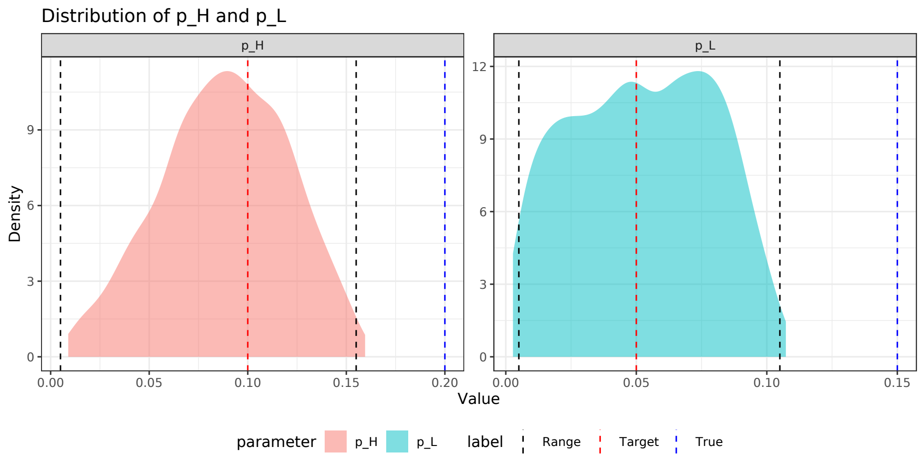


Fig. 3: Crumbly Cliff Walk inverse problem results. The red area shows the distribution of sampled values of $\tilde{p}^H$ from several runs of the algorithm, and the blue is the same for $\tilde{p}^L$. The blue lines show the probabilities under which the data was generated, the red lines are pre-selected values of the probabilities that make $\tilde{\pi}$ optimal, and the black lines are the true range of solution probabilities. In this case, $\tilde{\pi}$ is defined as the risky policy.

represented by the black lines. We note that, despite the lack of any theoretical guarantees, the algorithm seems to explore the true range well.

Since this same simulation study can be run for 6 different variations of the problem, from each of the three policies to the other two, we will not put all the images here. Instead, they can be found in Appendix C.

6 FOURTH DOWN PROBLEM

As a motivating example, we explore the case where two agents, with data from the same environment, appear to use very different optimal policies. For example, consider fourth down decision making in the NFL, and the fact that NFL Coaches, who are experts in their field, frequently do not make the statistically optimal decision in fourth down situations [12, 16, 11]. Figure 1 illustrates this difference between the typical coach policy and the statistically optimal policy. Data for this section comes from the *nflreadr* package [8], which is available for use under the MIT license.

[13] explored a very similar problem, using inverse optimization to understand coach fourth down decision making. They took the approach of estimating the quantiles of the empirical transition dynamics that best explain coach actions. This describes how risk-averse they are, relative to the empirical transition probabilities. Our method, by directly estimating what transition probabilities

coaches believe with uncertainty, gives us a more granular view of the difference between the observed and optimal actions.

6.1 Problem Setup

The details of our fourth down environment specification can be found in Appendix B. After defining the components of the MDP, we use weighted multinomial logistic regression to estimate the transition probabilities from each fourth down state to the next scoring state, as described in [17], then use these probabilities to estimate the *action-value* function $\hat{q}_{\pi_{\mathbf{w}}}(\sigma, a)$.

Rather than estimating $\tilde{\pi}$ and comparing it to an estimated optimal policy, we then compare the history of observed actions $\mathbf{a} = (a_1, ..., a_N)$ and the corresponding states $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_N)$ to estimated optimal actions $\mathbf{a}^*(\boldsymbol{\sigma}, \hat{q}_{\pi_{\mathbf{w}}}(\sigma, a)) = (a_1^*(\sigma_1, \hat{q}_{\pi_{\mathbf{w}}}(\sigma, a)), ..., a_N^*(\sigma_N, \hat{q}_{\pi_{\mathbf{w}}}(\sigma, a)))$, with $a_t^*(\sigma_t, \hat{q}_{\pi_{\mathbf{w}}}(\sigma, a))$ defined as

$$a_t^*(\sigma_t, \hat{q}_{\pi_{\mathbf{w}}}(\sigma, a)) = \underset{a \in \mathcal{A}}{\arg\max} \quad \hat{q}_{\pi_{\mathbf{w}}}(\sigma, a), \tag{9}$$

with $\hat{q}_{\pi,\mathbf{w}}$ as defined in Equation (14). We then compare using some loss function $\ell(\cdot, \cdot)$ as follows

$$\min_{\mathbf{w}} \ell(\mathbf{a}, \mathbf{a}^*(\boldsymbol{\sigma}, \hat{q}_{\pi_{\mathbf{w}}})). \tag{10}$$

We use the misclassification rate as our loss function, as defined in Equation (17) and Algorithm 1 to obtain solution weights.

6.2 Results

Expected Points Shift The simplest way to interpret the algorithm's results is to drill down on a specific field position. For example, the next plot shows the empirical and reweighted distributions from a single run of the algorithm for each of the actions when between the 31 and 40 yardlines with 3 or 4 yards to go. We can see that the average return of Going for it is the highest empirically, but the field goal is better after reweighting. The punt results are unimportant, since it is not often done in that situation. Interestingly, the algorithm only puts weight on outcomes with a reward of 3 for the field goal attempt, indicating that coaches might overestimate the probability of a successful field goal.

Quantifying Bias As we have discussed, one of the main strengths of our algorithm is the capability to quantify uncertainty in a way. While we are able to quantify the uncertainty in the transition probabilities themselves, the space is far too large to visualize. Instead, we will explore the uncertainty in the estimated values from those probabilities, stratified by action, location on the field, and yards to go, as visualized in Figure 5.

To do this, we run the algorithm 100 times, obtaining 100 different samples from the solution space. We then compare the optimal value function $\hat{v}$ and the believed value function $\tilde{v}$ with the difference $\tilde{v} - \hat{v}$ for each observed solution.

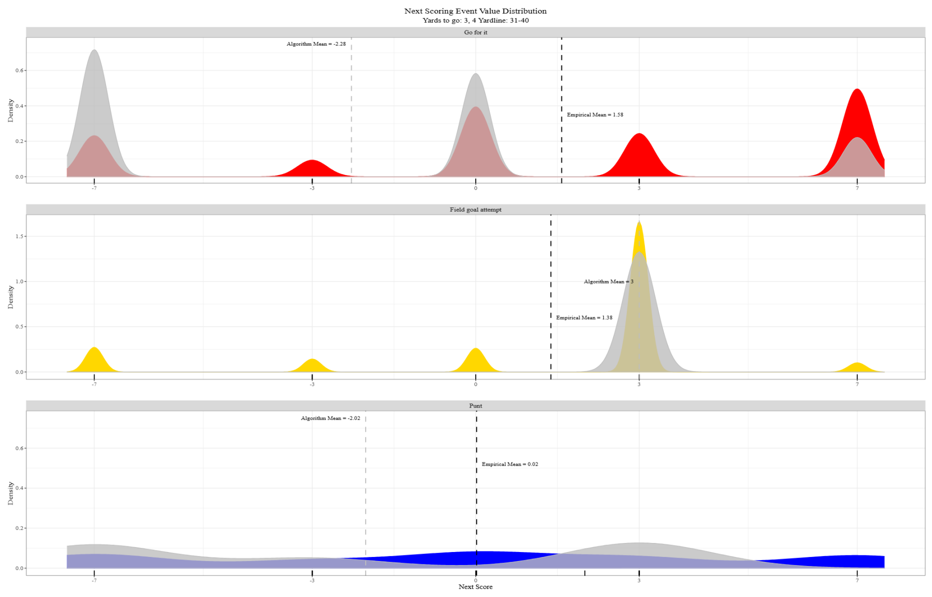


Fig. 4: Empirical and weighted density plots of the three actions when between the 31 and 40 yardlines with 3 or 4 yards to go. Colored densities show the empirical distribution, while the grey shows the reweighted distribution.

Note, a positive value means the agent overvalues that state, and vice versa. Then, with the list of differences, we take the range and mean values, as shown in Figure 5. We can see that there is a large region in the Go for it section that is blue over the entire range, indicating that there is strong evidence coaches undervalue going for it in those situations.

Other results can be found in Appendix B.

7 DISCUSSION

This paper introduced Importance Learning, a data-driven framework for the inverse Markov decision process when the reward function is known and transition probabilities are unknown. By reweighting observed transitions, our method infers the transition dynamics that make a decision maker's observed behavior minimally suboptimal. This provides an interpretable way to quantify how decision-makers implicitly distort or emphasize past experiences when forming beliefs about their environment.

The application to the fourth down decision problem shows that this approach can reveal meaningful behaviorial patterns consistent with human biases. The learned weights offer a measure of each observation's influence on perceived dynamics, yielding a novel quantitative window into biased decision-making.

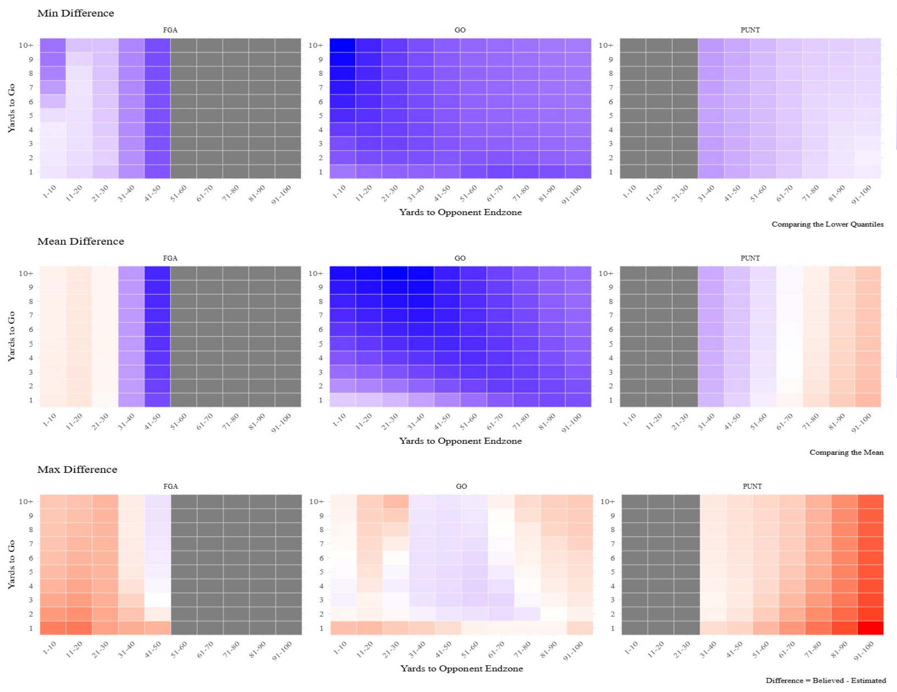


Fig. 5: Range of the estimated value functions obtained from 100 runs of Algorithm 1. States are visualized by how many yards to go for a first down and binned by position on the field. Positive values in a cell indicate that coaches overvalue that action in that state, and negative values are the opposite.

The main limitation is that Importance Learning is heuristic. While the method performs well on the examples we have explored, we provide no formal guarantees of convergence, identifiability, or uniqueness. Formal proofs of theoretical properties, such as existence, consistency, or asymptotic behavior of the learned weights, are important directions for future work.

Acknowledgments. A bold run-in heading in small font size at the end of the paper is used for general acknowledgments, for example: This study was funded by X (grant number Y).

Disclosure of Interests. It is now necessary to declare any competing interests or to specifically state that the authors have no competing interests. Please place the statement with a bold run-in heading in small font size beneath the (optional) acknowledgments[4], for example: The authors have no competing interests to declare that are relevant to the content of this article. Or: Author A has received research

[4] If EquinOCS, our proceedings submission system, is used, then the disclaimer can be provided directly in the system.

grants from Company W. Author B has received a speaker honorarium from Company X and owns stock in Company Y. Author C is a member of committee Z.

Bibliography

[1] Ahuja, R.K., Orlin, J.B.: Inverse optimization. Operations research **49**(5), 771–783 (2001)

[2] Baldwin, B.: nfl4th. https://www.nfl4th.com/articles/articles/4th-down-research.html (2021), accessed: 2021-03-31

[3] Burke, B., Carter, S., Giratikanon, T., Quealy, K., Daniel, J.: 4th down: When to go for it and why. https://www.nytimes.com/2014/09/05/upshot/4th-down-when-to-go-for-it-and-why.html (2014), accessed: 2021-03-23

[4] Chan, T.C., Fernandes, C., Puterman, M.L.: Points gained in football: Using markov process-based value functions to assess team performance. Operations Research (2021)

[5] Chick, C., Pardo, S., Reyna, V., Goldman, D.: Decision making (individuals). In: Ramachandran, V. (ed.) Encyclopedia of Human Behavior (Second Edition), pp. 651–658. Academic Press, San Diego, second edition edn. (2012). https://doi.org/https://doi.org/10.1016/B978-0-12-375000-6.00122-1, https://www.sciencedirect.com/science/article/pii/B9780123750006001221

[6] Erkin, Z., Bailey, M.D., Maillart, L.M., Schaefer, A.J., Roberts, M.S.: Eliciting patients' revealed preferences: An inverse markov decision process approach. Decision Analysis **7**(4), 358–365 (2010)

[7] Ghatrani, Z., Ghate, A.: Inverse markov decision processes with unknown transition probabilities. IISE Transactions **55**(6), 588–601 (2023)

[8] Ho, T., Carl, S.: nflreadr: Download 'nflverse' Data (2025), https://nflreadr.nflverse.com, r package version 1.5.0

[9] Kahneman, D., Tversky, A.: Prospect theory: An analysis of decision under risk. In: Handbook of the fundamentals of financial decision making: Part I, pp. 99–127. World Scientific (2013)

[10] Puterman, M.L.: Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons (2014)

[11] Roach, M.A., Owens, M.F.: Updating beliefs based on observed performance: Evidence from nfl head coaches. Journal of Sports Economics p. 15270025231222633 (2024)

[12] Romer, D.: Do firms maximize? evidence from professional football. Journal of Political Economy **114**(2), 340–365 (2006)

[13] Sandholtz, N., Wu, L., Puterman, M., Chan, T.C.: Learning risk preferences in markov decision processes: An application to the fourth down decision in the national football league. The Annals of Applied Statistics **18**(4), 3205–3228 (2024)

[14] Spranca, M., Minsk, E., Baron, J.: Omission and commission in judgment and choice. Journal of experimental social psychology **27**(1), 76–105 (1991)

[15] Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT press (2018)

[16] Yam, D.R., Lopez, M.J.: What was lost? a causal estimate of fourth down behavior in the national football league. Journal of Sports Analytics **5**(3), 153–167 (2019)
[17] Yurko, R., Ventura, S., Horowitz, M.: nflwar: a reproducible method for offensive player evaluation in football. Journal of Quantitative Analysis in Sports **15**(3), 163–183 (2019)

A Proof of Proposition 1

Proof. For a fixed policy π, transition model p, and reward function R, the value function v_π^p is uniquely determined as the solution to the Bellman equation:

$$v_\pi^p(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|s,a)\big[r(s,a,s') + \gamma v_\pi^p(s')\big].$$

Define the *effective transition kernel* induced by π:

$$p_\pi(s'|s) := \sum_a \pi(a|s) p(s'|s,a),$$

and the *policy-conditioned reward*:

$$r_\pi(s,s') := \frac{\sum_a \pi(a|s)p(s'|s,a)r(s,a,s')}{p_\pi(s'|s)}.$$

Then the Bellman equation can be rewritten as

$$v_\pi^p(s) = \sum_{s'} p_\pi(s'|s)\big[r_\pi(s,s') + \gamma v_\pi^p(s')\big].$$

Hence, v_π^p depends only on p_π, the *effective transition dynamics* under π, and not on the individual components $p(s'|s,a)$.

Consequently, any two transition models p_1 and p_2 satisfying

$$\sum_a \pi(a|s)p_1(s'|s,a) = \sum_a \pi(a|s)p_2(s'|s,a), \quad \forall s, s',$$

induce the same effective transition kernel p_π, and therefore the same value function v_π^p. Since the optimality of π depends only on the relative values $v_\pi^p(s)$ and $v_{\pi'}^p(s)$, all such transition models p will make π optimal.

Define the inverse set of transition models consistent with π as

$$\mathcal{I}_R(\pi) := \{\, p \mid \pi \in \arg\max_{\pi'} v_{\pi'}^p \,\}.$$

Then $\mathcal{I}_R(\pi)$ may contain infinitely many elements whenever distinct p's induce the same p_π.

As a simple counterexample, consider a two-armed bandit with actions $\{A, B\}$ and a deterministic policy $\pi(a = A) = 1$. Here, any set of transition probabilities leading to expected returns satisfying $v_A > v_B$ makes π optimal. Different transition models p can yield this inequality, so multiple p correspond to the same optimal policy.

Therefore, the transition model p is *not identifiable* from (R, π): the dependence of v_π^p on p occurs only through the product $\pi(a|s)p(s'|s, a)$, so π and p cannot be separated.

B APPLICATION TO THE NFL FOURTH DOWN DECISION (long)

As a motivating example, we explore the case where two agents, with data from the same environment, appear to use very different optimal policies. For example, consider fourth down decision making in the NFL, and the fact that NFL coaches, who are experts in their field, frequently do not make the statistically optimal decision in fourth down situations [12, 16, 11]. Figure 1 illustrates this difference between the typical coach policy and the statistically optimal policy. Data for this section comes from the *nflreadr* package [8], which is available for use under the MIT license.

[13] explored a very similar problem, using inverse optimization to understand coach fourth down decision making. They took the approach of estimating the quantiles of the empirical transition dynamics that best explain coach actions, describing how risk-averse they are relative to the empirical transition probabilities. Our method, by directly estimating what transition probabilities coaches believe with uncertainty, gives us another angle to explain the difference between their actions and the optimal actions.

B.1 The Fourth Down Decision as an MDP

We can mathematically describe fourth down behavior in terms of as MDP, defined in Section 1. However, now we can be more specific about the components of that MDP.

States We define the state space, $\mathcal{S}$, to be the union of two mutually exclusive sets: $\mathcal{S}^{\text{score}}$, the set of scoring states, and $\mathcal{S}^{\text{play}}$, the set of game states at the start of a play. A scoring state is a tuple consisting of the team that scored (denoted generically as either team A or B) and the type of score (touchdown, field goal, safety, or No Score):

$$\mathcal{S}^{\text{score}} = \{\text{A}, \text{B}\} \times \{\text{TD}, \text{FG}, \text{SAF}, \text{NS}\}. \tag{11}$$

A play state is a tuple consisting of the team with possession, down, yardline, yards to go until first down, seconds left in the half, the score differential (possession team's score - opponent team's score), and whether or not it is the fourth quarter:

$$\mathcal{S}^{\text{play}} = \{\text{A}, \text{B}\} \times \{1, 2, 3, 4\} \times [1, 99] \times \\ \{1, \ldots, 9, 10+\} \times [1, 1800] \times (-\infty, \infty) \times \{0, 1\}.$$

Let $\mathcal{S}^4 \subseteq \mathcal{S}^{\text{play}}$ denote the set of fourth down play states. Going forward, elements of $\mathcal{S}^4$ will be denoted σ.

Actions Given that we treat future actions as deterministic according to fixed policy π, the only action set we are concerned with is the set of fourth down actions. Hence $\mathcal{A} = \{\text{GO}, \text{FGA}, \text{PUNT}\}$, denoting go for it, field goal attempt, and punt, respectively. The set of realistic options for a given fourth down situation is typically either $\{\text{GO}, \text{FGA}\}$ or $\{\text{GO}, \text{PUNT}\}$, depending on the field position. We use $\mathcal{A}$ for simplicity.

Reward Function We define the reward function from the perspective of team A:

$$r(s) = \begin{cases} 7, & s = (\text{A, TD}) \in \mathcal{S}^{\text{score}}, \\ 3, & s = (\text{A, FG}) \in \mathcal{S}^{\text{score}}, \\ -2, & s = (\text{A, SAF}) \in \mathcal{S}^{\text{score}}, \\ 0, & s = (\text{A, NS}) \in \mathcal{S}^{\text{score}}. \end{cases} \tag{12}$$

If team B scores, the reward values are the negative of the values in Equation (12). The reward state of NS only occurs if no other scoring state is observed before the end of the half. [5] For simplicity, we model the reward of a touchdown as being worth 7 points. Previous studies ([4], [12], and [17]) have found that the league-average value is close to this, and we don't wish to include the complexity of one- and two-point conversion attempts in this work.

Objective Function In a fourth down situation, the coach wants to choose the action that will maximize the long term reward, assuming they will follow a given policy π thereafter. Thus, the optimization problem will be of the form

$$\max_{a \in \mathcal{A}} q_\pi(\sigma_t, a) \tag{13}$$

where $q_\pi(\sigma_t, a)$ represents the value associated with taking action a in fourth down state σ at time t. In our analysis, we will fix π at the *league-average policy* $\bar{\pi}$, which we assume to be stationary. We will use $q_{\bar{\pi}}(\sigma_t, a)$ as defined in (**??**) with the rewards $r(s)$ as defined in (12).

B.2 Estimating the Objective Function

In order to estimate the objective function based on our dataset, we will use logistic regression. Specifically, for each fourth down play σ in our dataset, we

[5] Since our state-space includes states that exclusively represent non-zero rewards (i.e., scores), we can define the reward function as depending on the state variable alone. This is unlike a typical MDP, in which the reward function may depend on both states and actions.

follow a similar procedure described by [17] to calculate the expected points in a given state given the action chosen from $\mathcal{A}$. To do this, we fit three separate logistic models on each of the scoring states conditioned on whether coaches decided to go for it, punt, or kick a field goal. Then these three models can be evaluated on a new state to determine whether going for it, punting, or kicking a field goal yields the highest expected points. We use this method because it effectively abstracts away the need to consider intermediate transition probabilities and the future policy. It also is simple to fit and evaluate on a new dataset, which is necessary to solve the inverse problem.

We define

$$\hat{\mathbf{p}}_{a,\sigma,\mathbf{w}} = \begin{bmatrix} \hat{p}(r = 7|\sigma_t, a, \mathbf{w}) \\ \hat{p}(r = 3|\sigma_t, a, \mathbf{w}) \\ \vdots \\ \hat{p}(r = -7|\sigma_t, a, \mathbf{w}) \end{bmatrix},$$

with $r = \sum_{k=t+1}^{K} r(s_k)$ and the probabilities estimated with weighted logistic regression using weights $\mathbf{w}$, and $v = \begin{bmatrix} 7 \; 3 \; \dots \; -7 \end{bmatrix}^T$ Then, we can estimate

$$\hat{q}_{\pi,\mathbf{w}}(\sigma_t, a) = \hat{\mathbb{E}}_{\pi} \left[\sum_{k=t+1}^{K} r(s_k)|\sigma_t, a, \mathbf{w} \right] = v^T \hat{\mathbf{p}}_{a,\sigma,\mathbf{w}} \tag{14}$$

for a given (a, σ_t) combination, then maximize over σ_t.

B.3 The Inverse Problem

For the fourth down problem, rather than estimating $\tilde{\pi}$ and comparing it to an estimated optimal policy, we will compare the history of observed actions $\mathbf{a} = (a_1, ..., a_N)$ and the corresponding states $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_N)$ to estimated optimal actions $\mathbf{a}^*(\boldsymbol{\sigma}, \hat{q}_{\pi,\mathbf{w}}) = (a_1^*(\sigma_1, \hat{q}_{\pi,\mathbf{w}}), ..., a_N^*(\sigma_N, \hat{q}_{\pi,\mathbf{w}}))$. We will define $a_t^*(\sigma_t, \hat{q}_{\pi,\mathbf{w}})$ as

$$a_t^*(\sigma_t, \hat{q}_{\pi,\mathbf{w}}) = \underset{a \in \mathcal{A}}{\arg\max} \quad \hat{q}_{\pi,\mathbf{w}}(\sigma_t, a), \tag{15}$$

with $\hat{q}_{\pi,\mathbf{w}}$ as defined in Equation (14) We will compare using some loss function $\ell(\cdot, \cdot)$ as follows

$$\min_{\hat{q}_{\pi,\mathbf{w}}} \ell(\mathbf{a}, \mathbf{a}^*(\boldsymbol{\sigma}, \hat{q}_{\pi,\mathbf{w}})). \tag{16}$$

The Loss Function Our goal will be to make $\mathbf{a}^*(\boldsymbol{\sigma}, q^{\bar{\pi}})$ as similar as possible to $\mathbf{a}$, the vector describing the decisions which coaches made in a given state. As such we will use accuracy as our objective, or one minus the accuracy as our loss function:

$$\ell(\mathbf{a}, \mathbf{a}^*(\boldsymbol{\sigma}, \hat{q}_{\pi,\mathbf{w}})) = \frac{1}{N} \sum_{t=1}^{N} \mathbb{I} \left[\mathbf{a}_t^*(\sigma_t, \hat{q}_{\pi,\mathbf{w}}) \neq \mathbf{a}_t \right]. \tag{17}$$

This loss function is not differentiable, but this is not a problem since Algorithm 1 does not require a differentiable loss. Unlike what we found in section 5, using a step function still works quite well since the fourth down dataset is very large.

B.4 Solving the Inverse Problem

As in Section 5, we will use Algorithm 1 to obtain solution transition probability functions to this inverse problem. We will make some slight alterations to the algorithm, however. Instead of specifying a loss threshold, we will allow the algorithm to run for some preset number of iterations, typically 100. This is to ensure that, when the algorithm is run many times, the completion time is predictable. However, in practice many of the runs obtained similar losses.

B.5 Results

Expected Points Shift The simplest way to interpret the algorithm's results is to drill down on a specific field position. For example, the next plot shows the empirical and reweighted distributions from a single run of the algorithm for each of the actions when between the 31 and 40 yardlines with 3 or 4 yards to go.

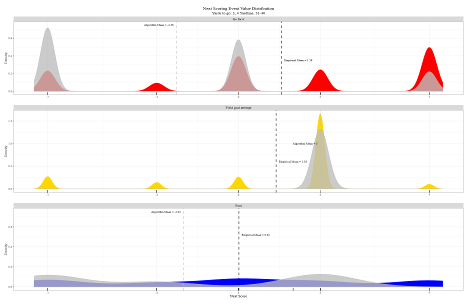


Fig. 6: Empirical and weighted density plots of the three actions when between the 31 and 40 yardlines with 3 or 4 yards to go. Colored densities show the empirical distribution, while the grey shows the reweighted distribution.

We can see that the average return of Going for it is the highest empirically, but the field goal is better after reweighting. The punt results are unimportant, since it is not often done in that situation. Interestingly, the algorithm only puts weight on outcomes with a reward of 3 for the field goal attempt, indicating that coaches might overestimate the probability of a successful field goal.

Comparison to Predictive Model To put our results in context, we fit a multinomial logistic model on the states with the observed coach actions as the response, then calculated the prediction accuracy of that model. It was able to achieve around 87.66% accuracy, which is lower than the 89.5% accuracy the algorithm was able to accomplish. This seems to indicate that our algorithm, by altering the MDP to make coach actions seem optimal, better explains coach behavior than a model that directly tries to predict what a coach would do.

Policy Visualization We can visualize broadly how our algorithm changes the suggested policy. In the following plot, the top left represents what coaches most commonly do in a given situation, and the top right represents what the expected points model recommends. Then, the bottom left represents the most common optimal action after reweighing the dataset with our algorithm. This third plot is visually much more similar to that of the observed policy, showing that our algorithm was generally successful in matching the observed actions.

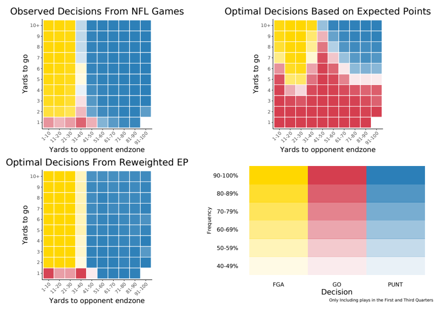


Fig. 7: Plot of three decisions maps representing average fourth down policies by location. Color indicates the most common action, with the alpha level being how frequent it is.

Visualizing Bias We can also visualize how biased coaches are in different locations on the field and for different actions. Figure 8 shows in the first row the average expected points in each location for each action based on the original data, and the second row shows the same for the reweighted data. We treat these reweighted expected points as what the coaches' believe. Finally, the third row is the difference between the two.

Additionally, taking advantage of the uncertainty quantification of our method, Figure 5 shows intervals for the difference at each location, as shown earlier.

From Figure 8, we can learn a couple of interesting things from the difference row. First, the go for it plot is mostly blue, indicating that the coaches undervalue going for it. Inversely, the punt plot is mostly red, indicating coaches overvalue punting it. Interestingly, the field goal plot is fairly white, which means that coaches seem to have correct beliefs about the outcomes of kicking the field goal.

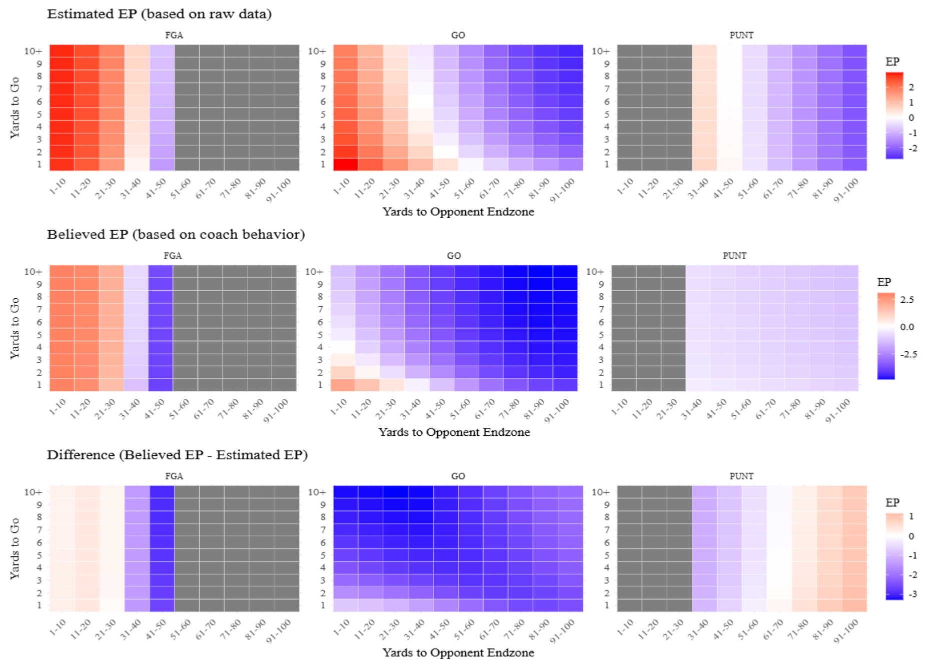


Fig. 8: Representations of Estimated Value Functions Based on 100 Runs of Algorithm 1. The Empirical Value Functions (top), Mean Believed Value Function (middle), and Believed-Empirical (bottom).

From Figure 5, we can confirm that some of the confidence intervals for the differences do not contain zero, suggesting that there are real differences. Unsurprisingly, go for it has the largest region of significant differences, confirming that the conclusions we drew from 8 are actually valid for those.

Bias over Time We can also explore coach bias over another aspect of the state space, time left in the game. Figure 9 shows curves representing the expected points (averaged over the rest of the state space) for both the "true", empirical model and the "believed", reweighted model. We also display another important benefit to our algorithm, the ability to quantify our uncertainty on the believed estimate. We do this by running the algorithm multiple times, and using the different resulting weights as samples from the sampling distribution of the weights. We can then get approximate confidence intervals on the believed estimate, and also estimate intervals on the empirical estimate with bootstrap techniques. This allows us to compare the estimates taking the uncertainty into account.

We can see that the Punt and Field goal estimates are largely overlapping, while the belief estimate for Going for it is almost entirely separate from the true estimate. This allows us to confidently conclude that there is evidence that

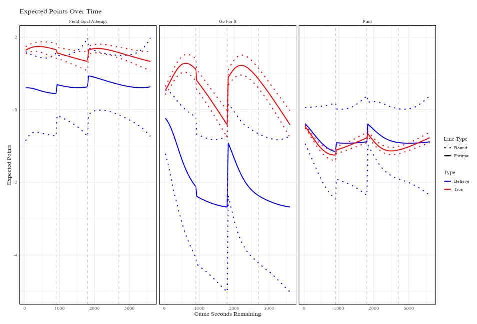


Fig. 9: Plots of the true (empirical) and believed (reweighted) average expected points estimates over the time dimension. Intervals show the range of observed values in the bootstrap process.

coaches have incorrect views about the value of going for it, at least over the Time dimension.

B.6 Discussion

These results make sense from both an intuitive and psychological perspective. We can easily observe that coaches often punt when they should go for it, so the result of undervaluing going and overvaluing punting makes sense. These are also consistent with psychological theories such as omission bias [14] and prospect theory [9].

In omission bias, decision makers seem to judge harm as a result of comission (action) more negatively than harm from omission (inaction). Going for it is a much more active decision while punting it is inactive, and both have a high potential of leading to negative outcomes. Thus, it follows that coaches view negative outcomes from going for it more harshly than punting it.

Prospect theory gives us another method to interpret coach behavior. In prospect theory, people tend to be risk averse for gains relative to the status quo, but risk seeking for losses relative to the status quo. For example, imagine that you were given \$2000, but you had to then choose between two options: losing \$1000 for sure or gambling on a 0.50 probability of losing \$2000 and a 0.50 probability of losing nothing. In response to this problem most people prefer to gamble (they are risk seeking). However, whereas most people prefer the gamble when faced with loss, they tend to prefer the sure option in the opposite scenario, when given the option between gaining \$1000 for sure or \$2000 with a 0.50 probability.[5] In the fourth down situation, coaches seem to be risk averse. This seems to indicate that coaches see their decision as choosing between a gain relative to the status quo. In other words, they are trying to make the decision that will give them the best outcome. However, it may be more logical for them to view their option as choosing between a loss. Most commonly on fourth down, coaches can either choose between punting it (losing \$1000 for sure, since the other team will definitely gain possession) or going for it (losing $2000 with some probability, since they may succeed but if they fail the put the other team in a better position). This insight may provide a way to communicate with coaches about the way they view decisions during the game and how they can change it to better act as analytics suggest.

C Cliff Walk Simulation Results

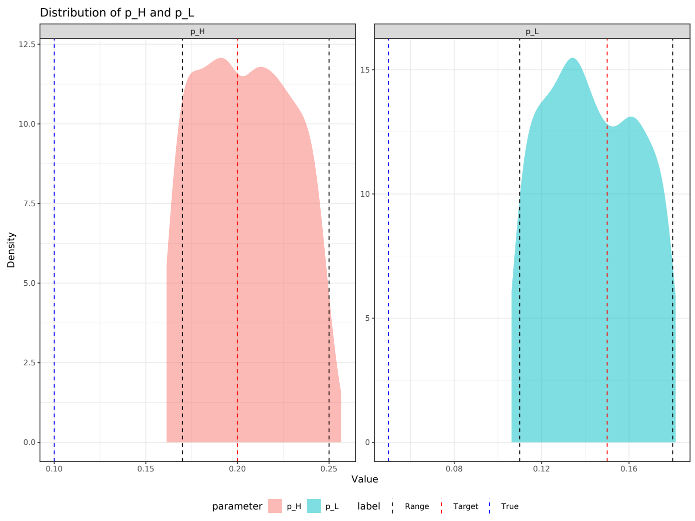


Fig. 10: Starting policy: risky, Target policy: safe

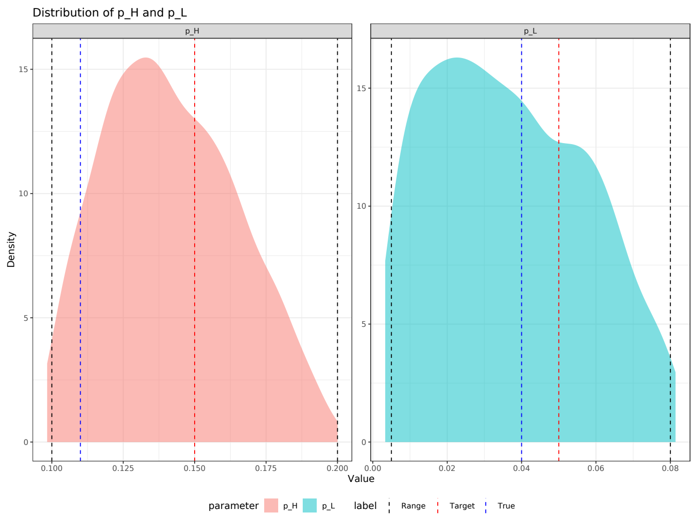


Fig. 11: Starting policy: risky, Target policy: safe

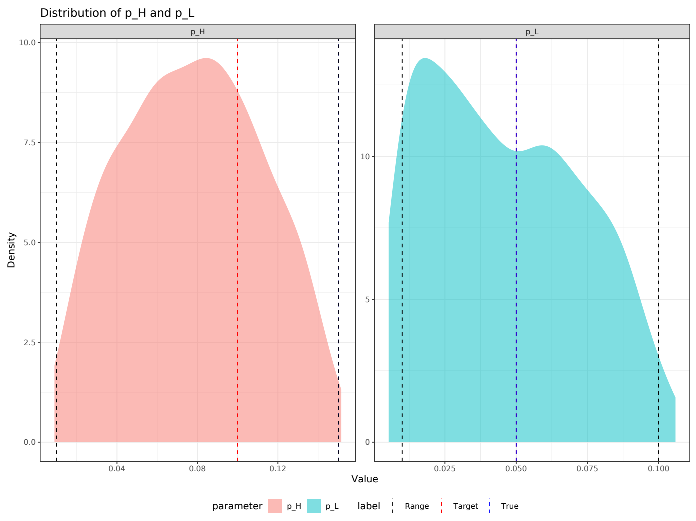


Fig. 12: Starting policy: medium, Target policy: risky

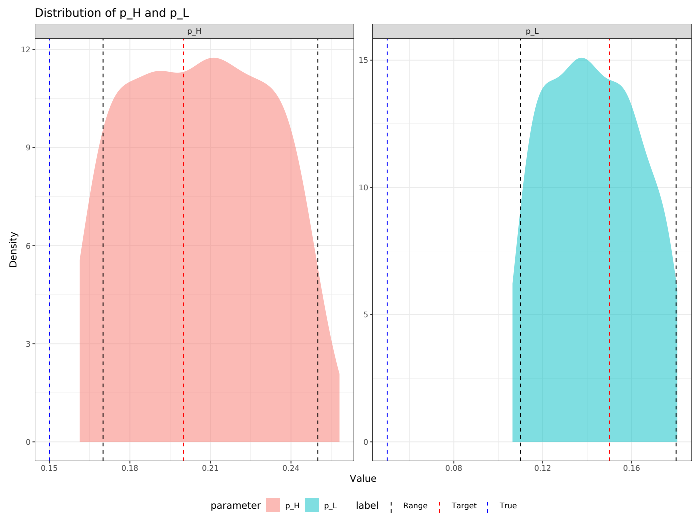


Fig. 13: Starting policy: medium, Target policy: safe

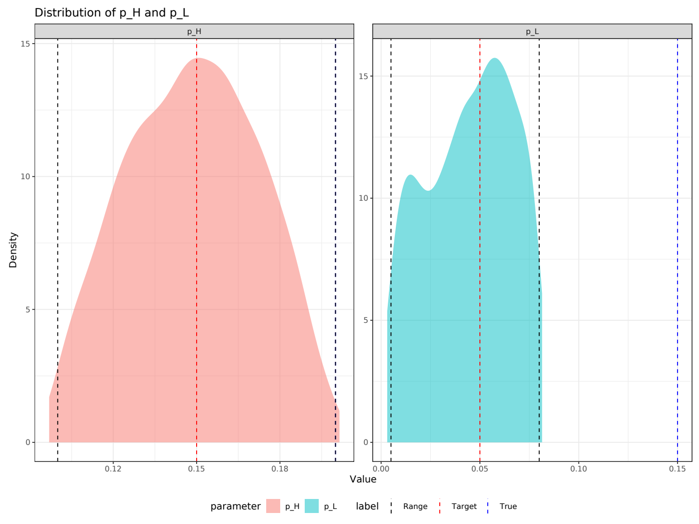


Fig. 14: Starting policy: safe, Target policy: medium

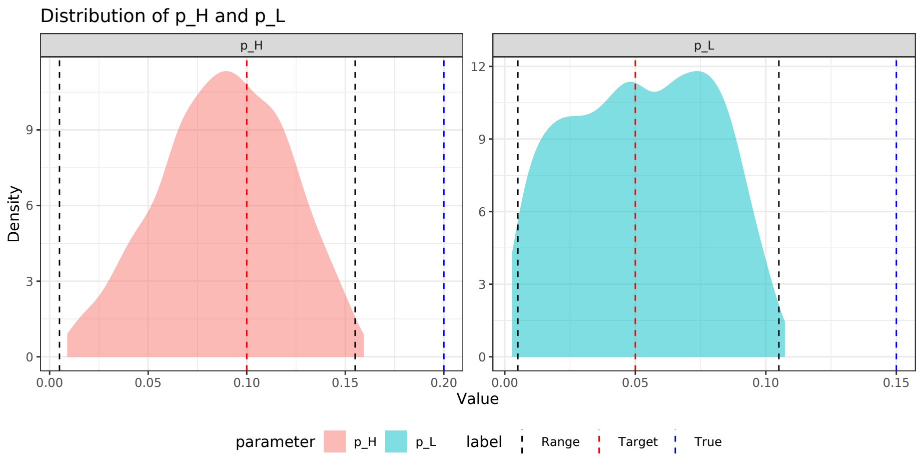


Fig. 15: Starting policy: safe, Target policy: risky