Structural Econometrics and Inverse Reinforcement Learning: Inferring preferences and beliefs from human behavior[*]

John Rust[†] Pranjal Rawat[‡]

June 30, 2025

PRELIMINARY DRAFT: NOT FOR CIRCULATION

Abstract

This article, for *Oxford Research Encyclopedia of Economics and Finance,* summarizes the literature on structural estimation of dynamic discrete choice models and points out close parallels to the rapidly growing literature on inverse reinforcement learning. Both literatures are focused on inferring underlying preferences (and sometimes beliefs) of human decision makers from observations of their states and actions.

[*] John Rust is grateful for support from his Georgetown University Professorship fund.

[†] Professor, Georgetown University jf693@georgetown.edu

[‡] PhD Candidate Georgetown Univeristy. pp712@georgetown.edu

1 Introduction

This article compares and contrasts methods developed in two different literatures for inferring preferences (and sometimes also beliefs) of human decision makers from observations on their behavior over time: one developed by economists that can be called "structural econometrics" (SE) and another developed by computer scientists in machine learning (ML) and artificial intelligence (AI) that is broadly under the general rubric called "reinforcement learning" (RL). Both SE and RL are huge literatures and we will not attempt to survey them in any breadth. Instead, we will focus on a particular subarea of SE known as *dynamic discrete choice* (DDC) which is most clearly parallel to a subareas of RL called "inverse reinforcement learning" (IRL).[1] Both literatures rely on the presumption that the agents whose preferences we are interested in recovering are *rational* i.e. they take a sequence of actions over time to maximize an internally perceived reward (or "utility") given their subjective beliefs of how their current actions cause or at least influence potentially uncertain future outcomes that affect their rewards. The goals of DDC and IRL are identical: to invert or infer their underlying objective or reward function (and sometimes also their beliefs) from observations of the sequence of states and actions of a sample of decision makers (DMs) or "agents" that could be individuals, firms, or other intelligent agents (e.g. animals). Beyond the intellectual interest in whether this type of inversion and inference is possible what are the practical motivations for DDC and IRL modeling?

In economics, structural models are used to test and evaluate economic theories, and to make *counterfactual predictions* of the effect of changes in economic policies or other changes in the environment on behavior and welfare (i.e. who is hurt and who is helped by the policy change?). Structural models are distinguished from reduced-form models that provide statistical summaries of agents' behavior, but do not attempt to infer their preferences and beliefs. Reduced-form models have limited ability to make counterfactual predictions of a policy change on behavior: this can be done by including policy variables as explanatory variables in the model. However when policies have not changed in the past, reduced form models cannot identify the effect of policy on behavior whereas structural models can produce accurate "out of sample" predictions of how the policy change affects behavior and welfare. For example, for decades Denmark has imposed one of the highest tax rates on new cars of any country in the world. Gillingham et al. (2022) used a structural model of equilibrium ownership and trading of cars in Denmark to predict the impact of reducing this "registration tax". The model predicts that by reducing the registration tax and increasing the gas tax, it would be possible of Denmark to 1) improve the welfare of its citizens, 2) increase total tax revenue from cars (registration plus gas tax), and 3) reduce total CO_2 pollution. It can be substantially faster and cheaper to use structural models to predict and evaluate the effects of various changes in economic policies than to actually carry them out and learning over time by trial and error. Auto producers learned this lesson decades ago when they developed finite element

[1] To space we will rely heavily on abbreviations throughout this article, but refer readers to table 1 for a summary of the various abbreviations and acronyms used throughout this article.

simulations of car crashes that were sufficiently realistic that they were adopted as a faster and cheaper way to evaluate new car designs. Increasingly realistic structural models are used to “crash test” policy changes in a similar way.

The practical motivation for inferring the DM’s reward function in IRL stems from the desire to produce intelligent agents that can perform tasks that involve goals that are complex, poorly-defined, or hard to specify, by inferring rewards underlying human performance of these same tasks, using limited human *demonstration data.* ML and RL have had astounding success in training machines to understand, predict and generate text, artwork, video, spoken language, and have made impressive strides in robotics. This success depends not only on advances in computer power and algorithms such as deep neural networks (DNN) that have the flexibility to make good predictions using a variety of unstructured inputs, but also on the existence of large amounts of training data, *and* a well defined objective function that the algorithm is being trained to optimize. As Russell and Norvig (2022) note, most of the existing ML methods use *supervised learning* where the algorithm “learns by passively observing example input/output pairs provided by a ‘teacher’.” (p. 840). So the reward function is known: predicting the choices of the teacher as well as possible. However the labelled data provided a by teacher is often quite limited, and this is where RL has proven so successful. When a reward function is known RL algorithms can produce behavior that approximately maximizes the reward function without an explicit teacher. For example Russell and Norvig (2022) note that in chess “there are relatively few examples (about 10^8) compared to the space of all possible chess positions (about 10^{40}).” When the reward function is known (e.g. in chess, 1 for winning and 0 for losing), RL has been able to learn superhuman peformance via *self-play,* i.e. repeatedly playing the algorithm against computerized opponents in “offline training” resulting in performance that exceeds the best human and computerized chess players in “online” real-time play, see e.g. Silver et al. (2016) and Silver et al. (2018).

But there are many challenging tasks in AI and robotics where it is unclear what reward function to use with RL to obtain intelligent behavior. Christiano et al. (2017) provide a concrete example: “suppose that we wanted to use reinforcement learning to train a robot to clean a table or scramble an egg. It’s not clear how to construct a suitable reward function, which will need to be a function of the robot’s sensors.” As AI starts to tackle increasingly complex, unstructured problems it becomes less clear what objective function should be used to train the algorithm using RL, and the option of “manually programming their behavior has become increasingly challenging and expensive.” Osa et al. (2018). The IRL literature seeks to overcome these problems by inferring a reward function from human choices and behavior on a task we would like to automate, but with access to only relatively small amount of human observations. If it is possible to estimate a reward function that “rationalizes” human choices, RL can be used train an algorithm to optimize the inferred reward function much more rapidly and cheaply, resulting in actions that are consistent with human preferences, and potentially enabling the algorithm to behave intelligently in situations that go far beyond the limited sample of data it was trained on.[2]

For example Barnes et al. (2024) used data on trips from Google Maps to “provide routes that best reflect travelers’ latent preferences” which can only be inferred from their “physical behavior, which implicitly trade-off factors including traffic conditions, distance, hills, safety, scenery, road conditions, etc.” They showed how to massively scale IRL to estimate a “route preference function” with 360 million parameters using a training dataset of 110 million trips sampled from Google Maps. Using this reward function, they used RL to train Google Maps to provide route recommendations more consistent with human preferences resulting “in a policy that achieves a 16-24% improvement in route quality at a global scale, and to the best of our knowledge, represents the largest published study or IRL algorithms in a real-world setting to date.”[3]

The remainder of this article is organized as follows. Section 2 provides background on dynamic programming (DP) and Markovian decision processes (MDP) which is the common ancestor of both the SE and RL literatures. We introduce the key concepts of value and policy function and the Bellman equation which characterizes the solution for an optimal dynamic policy, and introduce RL as a class of iterative stochastic algorithms for solveing the MDP problem. Section 3 describes how the DP framework is adapted to inference for the reward function in the SE literature, focusing on the *dynamic discrete choice* (DDC), and deriving the equations for inference of preferences and beliefs by the method of maximum likelihood. We also discuss the *identification problem* that puts inherent limits on our ability to uniquely recover the true preferences and beliefs of a decision maker in the absences of strong additional assumptions on their functional form. Section 4 summarizes the literture on IRL and its relation to the DDC literature. Finally, Section 5 concludes.

2 Background on DP and MDPs

The SE and RL literatures both build on a body of work that provides a recursive characterization of optimal sequential decision making under uncertainty, commonly referred to as *dynamic programming,* (DP) the name coined by one of the prominent early contributors to this theory, Bellman (2017). We will briefly review a special class of DP problems known as stationary, infinite horizon Markovian Decision Problems (MDP) introduced by Bellman and Howard (1960) that is

[2] There is also a literature on “inverse optimal control” (IOC) originated by Kalman (1964) that is also related to structural estimation and IRL. One of the early practical successes in this literature, Mombaur et al. (2010), was in robotics where IOC was used to “generate natural overall locomotion trajectories from an initial rest position an orientation to a given target rest position and orientation.” Abbeel (2016) extend the IOC approach to use DNNs to provide more flexible approximations to the underlying reward or cost function that “can learn complex, nonlinear cost representations . . . and can be applied to high-dimensional systems with unknown dynamics” resulting in a method that “outperforms prior IOC algorithms on a set of simulated benchmarks, and achieves good results on several real-world tasks.” See Azar et al. (2020) for further connections between IOC and IRL.

[3] While 110 million observations may not seem “limited”, it is a tiny fraction of all trips navigated using Google Maps. Jeon et al. (2021) cites additional examples where “defining a reward function beforehand is particularly challenging and IRL is simply more pragmatic.” These examples include robotic maniuplation, autonomous driving, and clinical motion analysis. See Zhao and Liang (2023) who used Adversarial IRL (AIRL) to estimate context-dependent route preferencs using data on trips by taxi drivers in Shanghai.

Table 1. List of abbreviations and acronyms

Abbreviation	**Full Name**
AI	Artificial Intelligence
AIRL	Adversarial Inverse Reinforcement Learning
AGI	Artificial General Intelligence
AL	Apprenticeship Learning
BC	Behavioral Cloning
BIRL	Bayesian Inverse Reinforcement Learning
CCP	Conditional Choice Probability
CCS	Conditional Choice Simulator
DDC	Dynamic Discrete Choice
DM	Decision Maker (Agent)
DNN	Deep Neural Network
DP	Dynamic Programming
DSE	Dynamic Structural Estimation
DU	Discounted Utility
EU	Expected Utility
IOC	Inverse Optimal Control
IL	Imitation Learning
IRL	Inverse Reinforcement Learning
MCMC	Markov Chain Monte Carlo
MDP	Markovian Decision Process
ML	Machine Learning
MPE	Markov Perfect Equilibrium
MPEC	Mathematical Programming with Equilibrium Constraints
NFXP	Nested Fixed Point
PI	Policy Iteration
RFE	Reduced Form Estimation
RL	Reinforcement Learning
RTDP	Real Time Dynamic Programming
SA	Successive Approximations
TD	Temporal Difference

the predominant underpinning of SE and RL and we refer the reader to Bertsekas (2017) for a first rate treatment of DP and MDPs.[4]

MDPs are defined by 1) a state space S, 2) an action space A, and 3) the objects $\{r, p, \beta\}$, where $r(s, a)$ is the *reward or utility function* providing the payoff to a decision maker (DM), $p(s'|s, a)$

[4] Note that DP and MDPs involve the optimization of discounted utility which was introduced by Samuelson (1937) combined with the expected utility concept introduced by von Neumann and Morgenstern (1944) for problems involving uncertainty. Though Samuelson "had concerns about its its descriptive realism, and it was never empirically validated as the appropriate model for intertemporal choice" (Frederick et al. (2002)), and laboratory tests including the famous *Allais Paradox* (see Machina (1987)) constitute evidence against the hypothesis of expected utility maximization, the MDP framework remains highly relevant for modern work in economics and AI and other areas because MDP model are sufficiently flexible to provide good approximations to a huge range of sequential decision problems. Though there is research that relaxes the discounted utility and expected utility assumptions, it leads to complications in the application of recursive methods to characterize and numerically approximate optimal strategies.

is a *transition density* for next period's state when the state is s and action is a and $\beta \in (0,1)$ is a discount factor. To simplify exposition we assume that both S and A are finite sets, with $|S|$ and $|A|$ elements, respectively.[5] The DM is presumed to choose a function $\pi(a|s)$ referred to as a *strategy, decision rule,* or *policy* that specifies the probability of taking action a in state s. For any $s \in S$ the support or π is restricted to a subset $A(s) \subset A$ of choices that are feasible for the DM to take in state s. Note that strategies can be either *mixed* (i.e. π is a probability) or *pure* (π is a function). The DM's objective is to choose a strategy π to maximize the expected dicounted stream of rewards under the policy π,

$$V(s) = \max_{\pi} V_\pi(s), \quad \text{where } V_\pi(s) = E_\pi \left\{ \sum_{t=0}^{\infty} \beta^t r(s_t, a_t) | s_0 = s \right\}, \tag{1}$$

where E_π denotes the expectation operator over the Markov process $\{s_t, a_t\}$ implied by the transition probability p and decision rule π. Bellman (2017), Howard (1960) and others proved that an optimal policy can be defined recursively in terms of what is now known as *Bellman's equation*

$$V(s) = \max_{a \in A(s)} \left[r(s,a) + \beta \sum_{s' \in S} V(s') p(s'|s,a) \right] \equiv \Gamma(V)(s), \tag{2}$$

so V is the fixed point $V = \Gamma(V)$ where the operator $\Gamma(V)$, known as the "Bellman operator", is the function of s implicitly defined by the right hand side of (2). The Bellman operator can also be shown to be a *contraction mapping* (so for any vectors $V, W \in R^{|S|}$ we have $\|\Gamma(V) - \Gamma(W)\| \leq \beta \|V - W\|$ where β is less than 1 and $\| \cdot \|$ is the Euclidean norm). This implies a unique solution V to the Bellman equation. The optimal policy can be recovered from V as the choice $a^* \in A(s)$ that attains the maximum in Bellman's equation

$$\pi(a|s) = \mathop{argmax}_{a \in A(s)} \left[r(s,a) + \beta \sum_{s' \in S} V(s') p(s'|s,a) \right]. \tag{3}$$

Note that equation (3) implies that the optimal policy is generically a pure strategy, i.e. except for rare case of ties, for each state s there is a unique $a^* \in A(s)$ that maximizes the right hand side of (3), so an optimal policy will choose this action with probability 1.

An MDP can be considered as a "game against nature" where "nature's move" is encoded in the transition probability $p(s'|s,a)$ and the optimal strategy π can be considered as a "best reply" to nature. This framework can be extended to dynamic games where there are multiple players in addition to nature. In this setting the reward and transition probability functions for each player can depend on the actions of their opponents. Maskin and Triole (2001) introduced the concept of *Markov Perfect equilibrium* (MPE) which is a generalization of the concept of Nash equilibrium

[5] All results below carry over essentially unchanged if S or A have an infinite number or even a continuum of elements provided certain boundedness and continuity conditions hold.

in static games where each player's strategy is a best reply to all of their opponents and nature. A MPE is defined by the solution to a *system of Bellman equations,* one for each player (except nature). Unlike the single Bellman equation in a single agent game against nature, this system of Bellman equations can have multiple solutions, corresponding to multiple MPE. Further, MPE strategies can be mixed whereas the optimal policy in a single agent MDP is generically pure. There are no known algorithms that are guaranteed to find even a single MPE, since unlike the single agent case, the system of Bellman equations is no longer a contraction mapping.[6] We now summarize several standard algorithms for solving the Bellman equation in the single agent setting.

2.1 Successive Approximations

As is well known, any contraction mapping on a Euclidean space has a unique fixed point, and the contraction property guarantees that the method of *successive approximations* i.e. the sequence $\{V_t\}$ given by $V_t = \Gamma(V_{t-1})$, for $t = 1, 2, \ldots$ starting from any initial starting guess V_0 will converge to the fixed point V. Successive approximations converge to V at a geometric rate in β and can be very slow when β is close to 1.

2.2 Policy Iteration

In many cases, a much faster method for solving the MDP is the *policy iteration* (PI) algorithm due to Howard (1960). This algorithm cycles between *policy valuation* and *policy improvement* steps. At iteration t of the PI algorithm there is a candidate policy $\pi_t(a|s)$. The policy valuation step calculates the value implied by this policy as $V_{\pi_t} \in R^{|S|}$ as the solution to the linear system

$$V_{\pi_t} = r_{\pi_t} + \beta E_{\pi_t} V_{\pi_t}, \tag{4}$$

where $r_{\pi_t} \in R^{|S|}$ is the vector whose s^{th} element is $r(s, \pi_t(s))$ and E_{π_t} is the $|S| \times |S|$ matrix whose with element in row s and column s' given by $E_{\pi_t}(s, s') = p(s'|s, \pi_t(a|s))$. It is easy to show that for any feasible policy π that E_π is a Markov transition probability matrix with matrix norm $\|E_\pi\| = 1$, which implies that equation (4) has a unique solution given by $V_{\pi_t} = [I - \beta E_{\pi_t}]^{-1} r_{\pi_t}$.[7] Given the value V_{π_t} the policy improvement step calculates a new policy π_{t+1} given by the right hand side of equation (3) except with V_{π_t} substituted for V. If $\pi_{t+1} = \pi_t$ then policy iteration has converged and it is not hard to show that $V_{\pi_t} = V$, the unique solution to Bellman's equation (2). Puterman and Brumelle (1979) showed that PI is mathematically equivalent to using Newton's method to find a zero of the function $F(V) = V - \Gamma(V)$, and as well known Newton's method is quadratically convergent. In fact, for discrete MDPs, PI converges in a finite number of iterations and typically the number of iterations is quite small, regardless of how close β is to 1. The main downside of PI is that for problems where the state space is large, the work required to solve the

[6] Iskhakov et al. (2015) introduced the *recursive lexicopgraphical search* (RLS) that can provably find all MPE in the subclass of dynamic directional games.

[7] Alternatively we can show that the policy-specific Bellman operator Γ_{π_t} in equation (4) is a contraction.

linear system (4) for V_{π_t} at each policy valuation step involves $O(|S|^3)$ operations. Thus, in the absence of sparsity or other "special structure" for the matrices E_{π_t}, alternative algorithms must be relied on for problems where the number of states $|S|$ exceeds several hundred thousand.

2.3 Avoiding the Curse of Dimensionality with function approximation

The computational burden of solving an MDP when the state space is large was recognized early on: Bellman referred to the problem as the *curse of dimensionality.* For example in problems where s is a continuous vector in R^d, if we approximate Bellman's equation using interpolation with a finite grid N points in each dimension, then $|S| = N^d$ and the total work to solve the MDP is $O(N^{3d})$, which increases exponentially fast in d. Even when the state space S is naturally discrete and finite, $|S|$ can be very large: for example in classic board games such as chess, the number of possible states (board positions) is estimated to be approximately 10^{44}. So it is clearly infeasible to use PI or even SA to try to solve MDPs.

Bellman and Dreyfus (1959) were among the first to recognize that one way to deal with the curse of dimensionality is to approximate the value function parametrically. For example we could represent it using a linear combination of fixed basis functions or approximate it with a neural network. Let ϕ denote the vector of parameters defining a parametric approximation to V with V_ϕ denoting the value function evaluated at parameter ϕ. Then an alternative way to approximate the solution to Bellman's equation is to convert it into a nonlinear least squares problem where $\hat{\theta}$ is chosen to minimize the squared "Bellman residuals" over a grid of points $(s_1, \ldots, s_J)$ in the state space, i.e.

$$\hat{\theta} = \underset{\phi}{argmin} \sum_{j=1}^{J} [V_\theta(s_j) - \Gamma(V_\phi)(s_j)]^2, \tag{5}$$

and the resulting value function $V_{\hat{\phi}}$ is an approximate solution to Bellman's equation, (2). We can also consider a modification of PI that we call *parametric policy iteration* (PPI), Benitez-Silva et al. (2000), where use a parametric approximation, $V_{\hat{\phi}_t}$ and find $\hat{\phi}_t$ to mimimize a least squares criterion similar to (25) to appxoimate the solution $V_{\pi_t} = r_{\pi_t} + \beta E_{\pi_t} V_{\pi_t}$ to each policy valuation step t, (4). When V_ϕ is approximated as a linear combination of "basis functions", $\hat{\phi}_t$ can be computed by linear regression rather than a nonlinear regression as in (25) when we replace the nonlinear Bellman operator Γ with the linear operator E_{π_t}. Most of the modern methods for solving high dimensional MDPs involve the use of function approximation, either for the value function or the policy function, or sometimes both simultaneously. We will discuss these methods in more detail in the section on reinforcement learning.

2.4 Reinforcement Learning

RL refers to a class of stochastic iterative algorithms for solving for the value function V and the optimal policy π. A prominent member of this class is called *Q-learning* introduced by Watkins

(1989) as a "general formal model of an animal's behavioural choices in its environment" and how it improves over time in response to feedback. In this sense Q-learning embodies a type of "learning" though for our purposes it can be viewed as a stochastic algorithm for solving MDPs since "Q" is just the choice-specific value function in an alternative way of expressing the Bellman equation

$$Q(s,a) = r(s,a) + \beta \sum_{s'} V(s')p(s'|s,a), \tag{6}$$

$$V(s) = \max_{a \in A(s)} Q(s,a). \tag{7}$$

Note that by substituting equation (7) into equation (6) we can represent Q as a fixed point $Q = \Lambda(Q)$ to a mapping Λ that can also be shown to be a contraction mapping. Thus, $Q(s,a)$ represents the value to taking action a in state s. The optimal policy chooses the action with the highest $Q(s,a)$ and the value function $V(s)$ equals the largest of the Q values in state s. Watkins proposed a stochastic iterative algorithm for updating Q given by

$$Q_{t+1}(s_t,a_t) = Q_t(s_t,a_t) + \alpha_t \left[r(s_t,a_t) + \beta \max_{a \in A(\tilde{s}_{t+1})} [Q_t(s_{t+1},a)] - Q_t(s_t,a_t) \right], \tag{8}$$

where (s_t,a_t) represent the state and action of the DM at iteration t, and $\tilde{s}_{t+1}$ is a random draw from the transition probability $p(s'|s_t,a_t)$, which implies that $\{Q_t\}$ is a stochastic process. The parameters α_t in the Q-learning iteration are called *step sizes* and must converge to 0 but at a sufficiently slow rate to guarantee the probability 1 convergence of the stochastic sequence $\{Q_t\}$ to $Q = \Lambda(Q)$ the unique Q function implied by the modified version of Bellman's in equations (6) and (7) as shown by Watkins and Dayan (1992) and Tsitsiklis (1994).[8]

Equation (8) does not specify how a_t is chosen: a typical choice is the *greedy policy* $a_t = argmax_{a \in A(s_t)} Q_t(s_t,a)$. However a key condition for the convergence of Q-learning is that every state-action pair (s,a) must be sampled an infinite number of times. This implies the usual exploration/exploitation tradeoff since suboptimal actions must be taken infinitely often for order for Q learning to converge, but this need not happen under a greedy policy. One solution to this problem is to use a ε*-greedy policy* that takes the optimal action a_t with probability $1-\varepsilon$, and with probability ε a randomly selected suboptimal action (i.e. an action a with $Q_t(s_t,a) < Q_t(s_t,a_t)$). However unless ε converges to 0 with t, Q_t will converge to a Q that does not satisfy the Bellman equation (13) since an ε-greedy policy is suboptimal.

Haarnoja et al. (2017) introduced *soft-Q learning* building on the work of Ziebart et al. (2008) on *maximum entropy reinforcement learning* that provides an alternative way to generate continuous exploration, ensuring that all actions are explored infinitely often. These result in policies that converge to mixed strategies that are self-consistent in the sense that the corresponding Q satisfy a modified version of the Bellman equation (2) they refer to as the *soft Bellman equation.* The term "maximum entropy" refers to the inclusion of an "entropy regularization" term in the agent's

[8] The conditions on the step size are 1) $\sum_t \alpha_t = \infty$ and 2) $\sum_t \alpha_t^2 < \infty$. Note $\alpha_t = 1/t$ satisfies these conditions.

objective function to result in optimal policies that are generically mixed. Specifically, suppose instead of the original objective in equation (1), we assume the agent's objective includes an entropy term, $H(\pi(s)) = -\sum_{a\in A(s)} \pi(a|s)\log(\pi(a|s))$, so the agent maximizes

$$V(s) = \max_{\pi} V_{\pi}(s), \quad \text{where } V_{\pi}(s) = E_{\pi}\left\{\sum_{t=0}^{\infty} \beta^t [r(s_t, a_t) + \sigma H(\pi(s_t))]|s_0 = s\right\}, \tag{9}$$

where σ is the weight on entropy. Note that when $\sigma = 0$ we obtain the original MDP objective (1) and the optimal π is generically pure and has 0 entropy, but as $\sigma \to \infty$ the agent's optimal policy is simply to choose each feasible action with equal probability, resulting in a π with maximum entropy. The solution V to the modified DP problem (9) turns out to satisfy a "soft Bellman equation" which turns out to be identical to a version of the "smoothed Bellman equation" derived by Rust (1988), see equations (14) and (13) of section 3. The economic interpretation is that randomness in actions does not stem from an inherent desire to maximize entropy, but rather due to the presence of idiosyncratic state variables (taste shocks) that affect the action the agent takes that make the optimal policy π appear random to someone who does not observe these taste shocks.

Q-learning is described as *model-free* since, unlike SA or PI, it does not require explicit calculation of expectations using the transition probability $p(s'|s, a)$: instead all that is needed is be able to *simulate new states using* p. In Watkins' example of how an animal learns to adapt and thrive in its environment, it would be the environment that "draws" the successor state s_{t+1} needed to implement the update in Q_t in equation (8). When trained in this manner Q-learning is indeed "model-free" and so are other RL algorithms that are trained "online" Barto et al. (1995) refer to these as *real time dynamic programming* (RTDP).

In many situations online Q-learning and RTDP produce policies that are too noisy and converge too slowly to be useful in practical real time applications, especially when the number of states is large. As Jiang and Xie (2024) note, their "decisions can lead to undesirable outcomes, especially at the early stages of learning when the algorithm has little knowledge of the environment. This is not a problem when the environment is a simulator, but can lead to serious consequences when people are part of the environment." As a result RL training is increasingly done *offline* with sufficient computer power to rapidly conduct many thousands or millions of training iterations to produce a much better solution than could be achieved from a limited amount on real-time, online experience. However when models are trained offline using computer simulations, they are no longer "model-free" since the computer needs the transition probability $p(s'|s, a)$ to make simulated draws of the successor states, s_{t+1}, needed by the Q-learning algorithm. The ability to simulate from the true $p(s'|s, a)$ (or a good approximation to it) and for sufficient number of iterations, are the key to producing good solutions using RL. In section 4 we return to this issue, showing that when we have sufficient data on state/action transitions in data on *past behavior* IRL can use this information to learn agents' rewards in an offline training in a way that is model-free.

3 Structural Estimation of DDCs

The development of new recursive methods for solving DPs and MDPs in the 1950s and 60s sparked a revolution in economics because these tools enabled economists to model sequential decision making under uncertainty, dynamic strategic interactions, and dynamic equilibria in markets which are key features of nearly all economic problems. Starting in the 1960s these tools facilitated the development of new dynamic economic theories that showed how expectations of the future affect the current choices of rational forward looking agents, resulting in new dynamic economic theories embodying *rational expectations* Muth (1961). Since this new framework had the flexibility to predict a wide range of behavior, the natural next step was to develop empirical methods for estimating and testing how well these new theories can explain data. This lead to a new area of econometrics that can be called *dynamic structural estimation* (DSE) which differed from existing econometric methods that could be classified as *reduced form estimation* (RFE). The term "structural" refers to the attempt to infer the underlying preferences and beliefs of economic agents rather than simply summarizing their behavior, the main goal of reduced form econometrics. In terms of the MDP framework, we can consider an agent's decision rule $\pi(a|s)$ to be the "reduced form model" because it captures their behavior, whereas structural econometrics tries to infer the underlying *structure* i.e. the preferences and beliefs $\{\beta, r, p\}$ as well as the implied optimal decision rule π.[9]

Robert E. Lucas (1976) provide a practical rationale for why we do structural econometrics. He argued that reduced-form econometric models produce misleading predictions of the effect of economic policy changes, whereas structural models that estimate the underlying preferences of consumers and technologies of firms in production of goods, are more likely to accurately predict the effects of policy changes. Policy evaluation can be regarded as making counterfactual predictions of how economic agents and the economy will react to changes in policy and the economic environment for which there is no historical antecedent, or with too little variation to be used as a covariate in a reduced form econometric model. To explain his argument in the MDP framework, let ρ devote a set of environment or policy variables that affect the preferences (or profits) and beliefs of economic agents. We can write the reward function as $r(s, a, \rho)$ and beliefs as $p(s'|s, a, \rho)$ to emphasize the dependence of these quantities on ρ. We can solve the MDP model for different values of ρ to to make counterfactual predictions of how behavior $\pi(a|s, \rho)$ and welfare $V(s, \rho)$ change in response to changes in ρ. As noted, when there is insufficient variation in ρ (or we want to consider changes in ρ beyond the range of historical experience), reduced form methods cannot provide good estimates of how $\pi(a|s, \rho)$ as ρ changes, nor can they predict how it affects welfare.

[9] The earliest development of structural econometrics dates back to work at the Cowles Foundation in thee 1940s where econometric methods were developed for inference on *linear simultaneous equation systems* which are static models of market equilibrium. In this work the structure was the coefficients of linear market supply and demand curves, whereas the reduced form were linear regressions that predict how equilibrium prices and quantities (p, q) shift with various variables x that shift the supply and demand curves. This was the earliest work showing how it could be possible to infer the structure (i.e. the coefficients of the supply and demand curves) using only data (p, q, x) on the intersections of the supply and demand curves in different market states x.

However it is sometimes possible to know how ρ affects $r(s,a,\rho)$ and $p(s'|s,a,\rho)$ even when there has been little or no variation in ρ in the past. For example consider the example of new car taxation in Denmark discussed in the introduction where ρ is the tax rate on new cars. Under the assumption that $r(s,a,\rho)$ additively separable in (s,a) and ρ Gillingham et al. (2022) used structural estimation to recover the component of $r(s,a,\rho)$ that depends only on (s,a), and with this, they could calculate $r(s,a,\rho)$ for any other possible tax rate ρ, due to the prior assumption on its functional form. In contrast we rarely know the functional form of $\pi(a|s,\rho)$ *a priori* but with knowledge of $r(s,a,\rho)$ and $p(s'|s,a,\rho)$ we can calculate $\pi(a|s,\rho)$ by solving the MDP. These solutions constitute counterfactual predictions of how car holding and trading behavior and welfare changes in response to changes in the new car tax rate ρ.[10]

The earliest work on estimation of dynamic structural models includes Sargent (1978), who estimated the dynamic linear quadratic model[11] of firms' demand for workers by maximum likelihood. In the remainder of this section we provide a selective summary of the literature on structural estimation of DDCs that has the closest connection to the literature on IRL. Key references include Rust (1987), Rust (1988), Rust (1994), Aguirregabiria and Mira (2002), and the survey by Aguirregabiria and Mira (2010).

We now consider how to infer the the structural objects $\{\beta, r, p\}$ from data on the trajectories of choices and states on a random sample of individuals using a modified version of the finite state and action MDP discussed in section 2. The modification is motivated by the *statistical degeneracy* of the optimal decision rule $\pi(a|s)$ in equation (3), i.e. it is generically a pure strategy (i.e. deterministic function of s). For most choices we actually observe, different individuals will choose different actions in the same observed state s. In order to explain this heterogeneity in behavior, we consider a modified MDP with an augmented state variable $x = (s,\epsilon)$ where the individual observes both s and ε (and thus uses a pure strategy) but we only observe s but not ε, which is a vector of idiosyncratic preference shocks that we do not observe. As a result, from our standpoint individuals appear to be using a mixed strategy. If there is sufficient variation in the unobserved ε component, we can derive a *conditional choice probability* (CCP) $\pi(a|s)$ from the overall (pure) strategy $\pi(a|x)$ that admits positive probabilities for all feasible choices a.

Building on the contributions of McFadden (1973) for static discrete choice models, we can derive an especially simple expression for the CCP $\pi(a|s)$ given by the *multinomial logit model* or "softmax" function in the RL literature. Suppose the choice sets $A(x)$ depend only on s and are finite, and assume that for each s, ε is a vector with the same length as $|A(s)|$ the number of actions, so for $\varepsilon(a)$ is the component of ε associated with action $a \in A(s)$ so the reward for action a in state a $r(x,a)$ can be written as $r(s,a) + \varepsilon(a)$. We call this the Additive Separability (AS) assumption. We also assume that the transition density $p(x'|x,a)$ for (x,ε) given a factors as

[10] Since changes in tax rates ρ aslo affect prices of new and used cars, a counterfactual prediction also has to account for how ρ affects the entire equilibrium in the Danish car market, so both new and used car prices are implicit functions of ρ. These equilibrium changes are also accounted for in the structural model of Gillingham et al. (2022).

[11] That is, the law of motion for s_t is a linear autoregression and r is a quadratic function of (s_t, a_t).

$f(\varepsilon'|s')p(s'|s,d)$, which we call the Conditional Independence (CI) assumption. Finally, we assume ε has a multivariate Gumbel or Type 1 extreme value distribution given by

$$F(\varepsilon|s) = \prod_{a \in A(s)} \exp\{-\exp\{-(\varepsilon(a) - \mu)/\sigma\}\}, \tag{10}$$

where we set $\mu = -\sigma\gamma$ where γ is Euler's constant, which implies that $E\{\varepsilon(a)|s\} = 0$ for $a \in A(s)$. We call this the Extreme Value (EV) assumption. Let $V(x) = V(s,\varepsilon)$ be the optimal value for this MDP, the solution to

$$V(s,\varepsilon) = \max_{\pi} V_\pi(s,\varepsilon), \quad \text{where } V_\pi(s,\varepsilon) = E_\pi\left\{\sum_{t=0}^{\infty} \beta^t [r(s_t,a_t) + \varepsilon_t(a_t)]|s_0 = s, \varepsilon_0 = \varepsilon\right\}, \tag{11}$$

Under assumptions AS, CI and EV, we can write the Bellman equation (2) for $V(s,\varepsilon)$ and show it has the representation

$$V(s,\varepsilon) = \max_{a \in A(s)} [Q(s,a) + \epsilon(a)], \tag{12}$$

where $Q(s,a)$ is given by

$$Q(s,a) = r(s,a) + \beta \sum_{s'} \sigma \log\left(\sum_{a' \in A(s')} \exp\{Q(s',a')/\sigma\}\right) p(s'|s,a). \tag{13}$$

The derivation of (12) and (13) makes use of an important property of the extreme value distribution for ε, namely that the expectation of the expectation of $V(s,\varepsilon)$ with respect to ε which an analytical expression, i.e. what is often called the log-sum or "smoothed max function" $V(s)$ given by

$$\begin{aligned} V(s) &\equiv E\left\{V(\tilde{s},\tilde{\varepsilon})\,|\tilde{s} = s\right\} = \int_\varepsilon \max_{a \in A(s)} [Q(s,a) + \varepsilon(a)]\, F(d\varepsilon|s) \\ &= \sigma \log\left(\sum_{a \in A(s)} \exp\{Q(s,a)/\sigma\}\right). \end{aligned} \tag{14}$$

Using equations (13) and (14) we can derive a "smoothed Bellman equation" for V

$$V(s) = \sigma \log\left(\sum_{a \in A(s)} \exp\left\{[r(s,a) + \beta \sum_{s'} V(s')p(s'|s,a)]/\sigma\right\}\right), \tag{15}$$

Equation (13) defines a "smoothed Q" as the fixed point of the smoothed contraction mapping $Q_\sigma = \Lambda_\sigma(Q_\sigma)$, and (15) defines V as a fixed point of the smoothed Bellman operator (also a contraction mapping) $V_\sigma = \Gamma_\sigma(V_\sigma)$. These operators are equivalent to the corresponding operators for MDP without the state variable ε in equations (6) and (7) and it is easy to show that Q_σ and V_σ converge to Q and V as $\sigma \downarrow 0$.

The representation of $V(x,\varepsilon)$ in equation (12) means that the decision rule $\pi(a|x,\varepsilon)$ is isomorphic to a static discrete choice model where $Q(s,a)+\varepsilon(a)$ is the utility/payoff from action a. McFadden (1973) showed that the CCP $\pi(a|s)$ is the conditional probability that the DM chooses action $a \in A(s)$ given the observed state s and has the classic multinomial logit formula given by

$$\begin{aligned}\pi(a|s) &= \int I\left\{Q(s,a)+\varepsilon(a) \geq \max_{a' \in A(s)} [Q(s,a')+\varepsilon(a')]\right\} F(d\varepsilon|s) \\ &= \frac{\exp\{Q(s,a)/\sigma\}}{\sum_{a' \in A(s)} \exp\{Q(s,a')/\sigma\}}. \end{aligned} \tag{16}$$

It is easy to show that as $\sigma \downarrow 0$ $\pi(a|s)$ converges to a degenerate probability that equals 1 if $Q(s,a)$ is the largest Q value and 0 otherwise, i.e. to the optimal pure strategy π in equation (3) for MDPs without the ε variable. Equation (16) can be considered as the generalization of the *policy improvement* step for MDPs in equation (3). Is there an analogous equation for the policy valuation step (4) for regular MDPs where the value for any policy π is the solution to a linear system? Aguirregabiria and Mira (2002) showed this does carry over, and the value V_π corresponding to any CCP π is given by

$$V_\pi(s) = \sum_{a \in A(s)} \pi(a|s)[r(s,a) + E\{\varepsilon(a)|s,a\} + \beta \sum_{s'} V_\pi(s')p(s'|s,a)], \tag{17}$$

where $E\{\varepsilon(a)|s,a\}$ is the conditional expectation of $\varepsilon(a)$ given that action a is chosen in state s under decision rule π. Under the extreme value assumption for ε, we have $E\{\varepsilon(a)|s,a\} = -\sigma \log(\pi(a|s))$. Using this, it is easy to see that the right side hand of (17) equals the smoothed max function defining the smoothed Bellman operator in equation (15). Thus, we can solve the smoothed Bellman equation $V_\sigma = \Gamma_\sigma(V_\sigma)$ using policy iteration just as in the case of MDPs, where we cycle between policy valuation steps (17) and policy improvement steps (16) using the Q_π implied by V_π in equation (13). In fact, just as in the case of MDPs, policy iteration is equivalent to Newton's method for solving the system $V_\sigma - \Gamma_\sigma(V_\sigma) = 0$.

Equation (16) implies that $V(s) = Q(s,a) - \log(\pi(a|s))$. Using this we can rewrite the nonlinear fixed point equation (13) for Q as a linear system depending on $r(s,a)$ and $\log(\pi(a|s))$,

$$Q(s,a) = r(s,a) + \beta \sum_{s'} \left[-\log(\pi(a'|s')) + Q(s',a')\right] p(s'|s,a), \tag{18}$$

where (18) holds for any choice of $a' \in A(s')$. It also implies that we can decompose Q as $Q(s,a) = Q_r(s,a) + Q_\varepsilon(s,a)$ where $Q_r(s,a)$ is the component of discounted value from observed rewards r and $Q_\varepsilon(s,a)$ is the component from unobserved rewards ε. Further substituting expression (17) in place of (15) in the equation for Q in (13), we can show that Q_r and Q_ε are each the solution to the following linear systems

$$Q_r(s,a) \quad = \quad r(s,a) + \beta \sum_{s'} \sum_{a' \in A(s')} Q_r(s',a')\pi(a'|s')p(s'|s,a) \tag{19}$$

$$Q_\varepsilon(s,a) \quad = \quad \beta \sum_{s'} \sum_{a' \in A(s')} \left[-\sigma \log(\pi(a'|s')) + Q_\varepsilon(s',a')\right] \pi(a'|s')p(s'|s,a). \tag{20}$$

This decomposition can be exploited to reduce the computational burden of estimating DDC models as we will show below.

3.1 Estimation Methods for DDCs

The most common estimation method for DDCs is some form of maximum likelihood. Suppose we observe a random sample of *trajectories* i.e. data on the consecutive states and actions taken by N individuals $\{(s_{it}, a_{it})|t = 0, \ldots, T_i,\ i = 1, \ldots, N\}$. Suppose that we smoothly parametrize (r, p) using a vector unknown parameters θ, which we denote by r_θ and p_θ.[12] Using the implicit function theorem, we can show that V_θ and Q_θ, the contraction fixed points given in equations (14) and (13), are continuously differentiable function of θ. Using these fixed points and the logit formula for $\pi(a|s)$ in equation (16) we can write a full likelihood function for the data as a function of the unknown parameters θ given by

$$L(\theta) = \prod_{i=1}^{N} \prod_{t=1}^{T_i} \pi_\theta(a_{it}|s_{it}) p_\theta(s_{it}|s_{it-1}, a_{it-1}). \tag{21}$$

The smoothness of V_θ and Q_θ in θ implies that the CCP π_θ and the likelihood $L(\theta)$ are both smooth functions of θ so the standard asymptotic efficiency properties of maximum likelihood apply.

Rust (1988) introduced the Nested Fixed Point algorithm (NFXP) to compute the maximum likelihood estimator (MLE) of θ. It consists of a standard outer hill climbing algorithm to maximize $L(\theta)$ over θ, but at each evaluation of θ it calculates the contraction fixed point $Q_\theta = \Lambda_\theta(Q_\theta)$ that defines the choice-specific values Q_θ where Λ_θ is the operator defined on the right hand side of equation (13). Notice that $L(\theta)$ depends on θ via Q_θ. Su and Judd (2012) proposed an alternative strategy for computing the MLE based on the MPEC algorithm that maximizes a likelihood $L(\theta, Q)$ expressed as a function of (θ, Q), subject to the fixed point constraint $Q_\theta = \Lambda_\theta(Q_\theta)$.[13]

Estimation of DDC models is often done in a two step fashion, where a parametric model of the transition probability $p(s'|s, d)$ it estimated in the first stage using a partial likelihood that does not involve the CCP π in (21), and in the second stage the parameters of r are estimated using

[12] footnotesize The discount factor β is often treated as known, though in some cases it is also estimated and thus part of θ.

[13] Su and Judd (2012) claim that the MPEC alogithm is substantially faster than NFXP, but Iskhakov et al. (2016) showed that this was due to using SA to compute the fixed point $Q_\theta = \Lambda_\theta(Q_\theta)$ which is slow when β is close to 1. When Newton's method is used to compute this fixed point, the cpu times for MPEC and NFXP are roughly comparable with NFXP being faster on test problems with large sample sizes.

another partial likelihood involving only π, treating the estimated parameters of p as if they were known. Often a final Newton step iteration is done using the full likelihood function (21) using the estimated parameters for p and r from steps 1 and 2 to obtain fully efficient parameter estimates with the correct asymptotic covariance matrix, i.e. it accounts for "first stage" estimation effort in the parameters of p.

The computational burden of NFXP stems from the need to compute the inner fixed point $Q_\theta = \Lambda_\theta(Q_\theta)$ for each trial value of θ in the outer hill-climbing algorithm that maximizes $L(\theta)$. Newton's method (e.g. policy iteration) is typically used to solve for Q_θ (or alternatively for V_θ in equation (17)). Even though Newton's method converges rapidly, it requires $O(|S|^3)$ operations per Newton/policy iteration step, making NFXP burdensome for large state spaces S. Therefore alternative estimation methods have been proposed that reduce the total number of policy iteration steps required to estimate θ. Aguirregabiria and Mira (2002) used the linear representation of the policy valuation step (17) to reverse the order of the outer maximization and inner fixed point steps of the NFXP algorithm resulting in what they call a *nested pseudo likelihood* (NPL) estimator that requires fewer total policy valuation steps than NFXP, and is as statistically efficient as NFXP when iterated to convergence. Dearing and Blevins (2025) proposed a similar iterative procedure called *efficient psuedo likelihood* (EPL) that is also an efficient procedure for estimating structural parameters of dynamic games.

Hotz and Miller (1993) proposed the "CCP estimator" for r that uses a non-parametric estimate of $\pi(a|s)$ to avoid repeated solution of Q over the search for θ that NFXP requires. Consider the case where r_θ is linear in parameters, i.e. we can write $r_\theta(s,a) = r(s,a) * \theta \equiv \sum_{k=1}^{K} r_k(s,a)\theta_k$ for known functions $\vec{r} = (r_1, \ldots, r_K)$ called *features* in the IRL literature. Then (19) implies that $Q_r(s,a) = R(s,a) * \theta$ where R is the solution to

$$R(s,a) = \vec{r}(s,a) + \beta \sum_{s'} \sum_{a' \in A(s')} R(s',a')\pi(a'|s')p(s'|s,a) = \vec{r}(s,a) + \beta ER(s,a), \tag{22}$$

where $ER(s,a)$ is a shorthand for the conditional expectation of $R(s,a)$ the middle of expression in (22). It follows that we only need to solve (22) once using a non-parametric estimate of $\hat{\pi}$ along with Q_ε in (20) to serve as a "correction term" to obtain the following form for the CCP

$$\pi_\theta(a|s) = \frac{\exp\{\hat{R}(s,a) * \theta + \hat{Q}_\varepsilon(s,a)\}}{\sum_{a' \in A(s)} \exp\{\hat{R}(s,a') * \theta + \hat{Q}_\varepsilon(s,a')\}}, \tag{23}$$

where $\hat{R}(s,a)$ is the solution to (22) and $\hat{Q}_\varepsilon$ is the solution to (20) using non-parametric first stage estimators $\hat{\pi}(a|s)$ and $\hat{p}(s'|s,a)$. The CCP estimator $\hat{\theta}$ maximizes the partial likelihood $L_\pi(\theta)$ given by

$$L_\pi(\theta) = \prod_{i=1}^{N} \prod_{t=1}^{T_i} \pi_\theta(a_{it}|s_{it}), \tag{24}$$

where π is given in (23).[14] Thus, the CCP estimator in the linear in parameters case requires only two solutions of systems with $|S|$ equations and unknowns. This can be further reduced to a single solution if we assume that $r(s, a_s)$ is known for some "normalizing alternative" $a_s \in A(s)$ as we discuss further below in the section on identification of DDC models.

Bayesian approaches have been proposed for inference in DDC models using Markov Chain Monte Carlo methods that simultaneously simulate the likelihood as well as stochastic simulations to compute estimates of value functions that are similar to "Monte Carlo tree search" used by RL algorithms such as the Q-learning algorithm discussed in section 2.4, see Imai et al. (2009) and Norets (2009). There are also non-Bayesian simulation estimators such as Bajari et al. (2007) that estimate θ as the value that minimizes the degree of suboptimality in observed actions relative to other actions not taken using simulation estimates of Q function evaluated at the actions the DM actually an other actions not chosen.

3.2 Function Approximation Methods for Large State Spaces

When the state space S is very large (i.e. the state space is continuous or the number of states $|S|$ is in the hundreds of thousands or higher), the all of the approaches discussed above that require one or more solutions of large linear systems with $|S|$ equations and unknowns become computationally infeasible.

An alternative approach is to use parametric methods to approximate the value function V as $V_\phi(s) = f(s, \phi)$ where $f(s, \phi)$ is a member of a flexible family such as neural networks where ϕ denote the weights (parameters) defining the network, and then recasting the fixed point problems (13) and (14) as potentially more tractable nonlinear least squares problems. For example we can approximate V as $V_{\hat{\phi}}$ where $\hat{\phi}$ are the network weights that solve the nonlinear least squares problem of minimizing the squared Bellman residuals in equation (25).

$$\rho(\theta, \phi) = \sum_{j=1}^{N} [V_\phi(s_j) - \Gamma_{\theta,\sigma}(V_\phi)(s_j)]^2. \tag{25}$$

Note that the smoothed Bellman operator Γ_σ defined on the right hand side of (14) depends on r and p which in turn are function of the model parameters θ, so we write it as $\Gamma_{\theta,\sigma}$. It follows that the minimizing parameters $\hat{\phi}$ are implicit functions of θ.

Let $\rho(\theta, \phi)$ denote the squared Bellman residuals defined over a grid of N points in the state space Then a "nested least squares" estimator similar to NFXP can be used estimate θ in large state spaces, where we maximize $L(\theta)$, but in the inner fixed point problem has been transformed into the problem of minimizing $\rho(\theta, \phi)$ over the parameters ϕ. $\rho(\theta, \phi)$ for each trial θ. The CCP estimator can also be implemented in a similar fashion where we parametrize R and Q_ε by parameters ϕ_R and ϕ_ε and solve two nonlinear least squares problems to approximate the fixed points in (22) and

[14] See Chernozhukov et al. (2022) for a modified version of the CCP estimator that is "locally robust" i.e. it corrects for sampling error in the first stage estimates of the functions $\hat{H}(s, a)$ and $Q_\varepsilon(s, a)$.

(20), respectively. Note, that these would only need to solved *once* at the start of estimation, and the resulting solutions $R_{\hat{\phi}_R}$ and $Q_{\hat{\phi}_\varepsilon}$ can then used to evaluate $\pi_\theta(a|s)$ in (23).

Luo and Sang (2024) introduced a penalized likelihood estimator where $\hat{\theta}$ and $\hat{\phi}$ are simultaneously chosen to maximize

$$(\hat{\theta}, \hat{\phi}) = \underset{\theta,\phi}{argmax} \log(L(\theta)) - \lambda\rho(\theta, \phi), \tag{26}$$

where $\lambda \geq 0$ is a penalization parameter. Note that the likelihood depends on both θ and ϕ since both the CCP π and Q are defined in terms of V_ϕ which depends on the network weights, ϕ, in addition to the structural parameters of interest, θ. The consistency of the penalized likelihood estimator requires the use of a *sieve,* i.e the dimension of ϕ must grow to infinity at certain rate with the sample size to guarantee that the error in approximating the true fixed point by V_ϕ converges to zero. As a result they refer to the estimator (26) as SEES, for Sieve-based Efficient Estimator for Structural models. See Peter Arcidiacono and James (2013) and Kristensen et al. (2021) for other examples of sieve-based estimators for DDC models.

3.3 The Identification Problem

At its core, structural estimation of DDC models can be regarded as a process of inverting observed behavior to uncover preferences. The observed behavior from the DDC model is captured by the CCP $\pi(a|s)$, and it referred to as a "reduced form" model of the agent's behavior since with enough data on (a, s) pairs, we can estimate π nonparametrically without any attempt to uncover the agent's preferences or beliefs by solving the MDP problem to derive π as outlined above. In econometrics this is referred to as reduced-form estimation (RFE), whereas in the ML literature it is referred to as "behavioral cloning" (BC) "imitation learning" (IM), or "apprenticeship learning" (AL). The analysis of identification treats π as known because with infinite data we could learn it perfectly. The preceding section showed how to derive π from the structure $\{\beta, r, p\}$, a process that we can summarize by the mapping $\pi = f(\beta, r, p)$. The identification problem concerns the invertibility of this mapping: given π is there a unique underlying structure $\{\beta, r, p\}$ that implies it? Unfortunately, without further restrictions the answer is negative, as shown by Hotz and Miller (1993), Rust (1994) and Magnac and Thesmar (2002). To understand why, consider the first step of the inversion process, uncovering Q from π. Using the logit form of π in (16) we have

$$\log\big(\pi(a|s)/\pi(a'|s)\big) = Q(s, a) - Q(s, a'), \quad s \in S,\ a, a' \in A. \tag{27}$$

We already see that information on π is insufficient to recover all the Q values: only the *differences* in Q are identified but not all $|S||A|$ possible values of $Q(s, a)$. Indeed, the structure $\{\beta, r, p\}$ contains $1 + |S|(|A| - 1) + |S|^2|A|$ parameters which is more than then $|S|(|A| - 1)$ parameters of π, so we will generally have "more equations than unknowns" so the underlying structure is not uniquely identified.

Thus, it is clear that in order to identification requires additional restrictions on $\{\beta, r, p\}$. For example we might impose the assumption that r and p have a parametric functional forms that depend on a relatively small number of parameters θ. For example Anderson et al. (2025) are able to identify the subjective beliefs of professional tennis servers about how their serve strategies affects their probability of winning a tennis game. These beliefs take the form of a transition $p(s'|s,a)$ where s denotes the point state of the game and a denotes serve direction. Identification is possible because it is reasonable to assume r is known: i.e. the server earns a reward of 1 if they win the game and 0 otherwise, and due to reasonable parametric restrictions on p.

In many applications researchers are willing to assume agents have rational expectations which implies that their subjective beliefs about state transitions $p(s'|s,a)$ correspond to the actual probability governing these transitions and so p can be estimated non-parametrically given sufficient data, and thus can be treated as known but with restrictions on r this assumption is testable.[15] Many studies typically assume that the discount factor β is also known, however under the assumption of rational expectations and restrictions on rewards, β can be identified, see Abbring and Daljord (2020).

Assuming $\{\beta, p\}$ are known, identification of r depends on the assumption that either it has a known parametric functional form r_θ where the dimension of θ is less than $|S|(|A|-1)$, or that for each $s \in S$ there is a normalizing or anchor action a_s such that $r(s, a_s)$ is known. The latter assumption amounts to $|S|$ restrictions that imply there are only $|S|(|A|-1)$ unknown rewards to be recovered, the same number of free values of π. Hotz and Miller (1993) and Kang et al. (2025) show that in this "exactly identified" case, we can calculate $Q(s, a_s)$ from knowledge of $r(s, a_s)$ and $\pi(a_s|s)$ using equation (18) and then (27) allows us to recover all remaining Q values. Using the Q values we can back out the remaining unknown rewards r from the smooth Bellman equation (13).

Note that the assumption that we know $r(s, a_s)$ for some anchor action a_s in each state $s \in S$ is far from innocuous. If this assumption is wrong, even though the DDC model has enough parameters to perfectly fit π, the estimated rewards will not equal the true rewards. In this case, counterfactual behavior predicted by the model with an incorrectly identified reward function can differ from what would actually happen under the true reward. Another way to say this is that in the absence of any prior restrictions on rewards, we can only identify a set of "obvservationally equivalent" rewards of the form $r(s,a) + h(s) - \beta \sum_{s'} h(s')p(s'|s,a)$ for any function $h : S \to R$ that is called the partially identified set. This constitutes an inherent limit on what can be learned about agents' rewards and emphasizes the importance of having accurate prior knowledge of at least some aspects of the reward function to obtain credible counterfactual predictions and welfare impacts of changes in policy and the environment. Kalouptsidi et al. (2021) provides examples of certain counterfactuals that are identified even when r is only partially identified. They conclude that their "results call for caution while leaving room for optimism: although counterfactual behavior and welfare can be sensitive to identifying restrictions imposed on the model, there exists important classes of counterfactuals that are robust to such restrictions." (p. 385).

[15] Anderson et al. (2025) strongly reject the hypothesis the many professional tennis servers have rational beliefs about their own ability and those of their opponent, as embodied by $p(s'|s,a)$.

4 Inverse Reinforcement Learning

The IRL literature was initiated by Ng and Russell (2000) who identified two motivations for inferring the underlying rewards that are presumed to generate observed behavior: 1) "the potential use of reinforcement learning and related methods as potential models for human and animal learning" and 2) to provide a reward function that RL can use to construct an "intelligent agent that can behave successfully in a particular domain." Though they note that imitation and apprenticeship learning (IL and AL) can be used to learn a policy π, "the reward function often provides a much more parsimonius description of behaviour" so that IRL can constitute an effective form of AL. IRL has become an important method in many areas of robotics including development of autonomous drones and vehicles (AV) where there is insufficient data on decisions to human drivers under multitudes of different road conditions to develop effective AVs. Wang et al. (2025) note that IRL has "found great success in predicting trajectories through inferring cost or reward functions from expert demonstrations, and then using these functions to guide the behavior of self-driving vehicles in unseen driving environments."

This section provides a selective review of the IRL literature focusing on methods that overlap most closely with the structural DDC literature reviewed in section 3. We refer readers to surveys by Arora and Doshi (2021), Gleave and Toyer (2022), and Adams et al. (2022) for a broader and more in depth view of IRL. The latter survey provides further examples of how IRL uses "an expert to demonstrate the successful task and then a highly parameterized reward function could be transferred to the robot for optimization" and how IRL can generate more effective policies than AL, particularly for new, counterfactual environments where "IRL has the added advantage that the reward function can be transferred to new environments with different dynamics.".

There are three broad classes of IRL algorithms: 1) max-margin, 2) Bayesian methods, and 3) maximum entropy methods. Ng and Russell (2000) introduced three different types of max-margin estimators, which search for a reward function for which payoffs under the policy of the observed expert are higher than any other policy (i.e. behavior), an approach that shares similarities with the "moment inequality" simulation estimator proposed by Bajari et al. (2007) discussed in section 3. They recognized the identification problem discussed in the previous section, namely there may be many reward functions that satisfy the max-margin inequalities (payoff inequalities) including the trivial solution $r = 0$.

Bayesian IRL (BIRL) was introduced by Ramachandran and Amir (2007). They assumed π is a mixed strategy with the softmax form (16) that depends on Q functions which are implicit functions or r per the Bellman equations (6) and (7) and interpret the σ parameter in the softmax function for π as controlling for the "degree of confidence we have in the [DM's] ability to choose actions with high value." This allowed them to formulate a likelihood for the sequence of observed actions by the DM that depends on r. Using Bayes' Rule, they computed a posterior distribution for r, treating it as the unknown parameter with a prior distribution. Since it is infeasible to directly compute the posterior, BIRL uses MCMC to generate draws from the posterior distribution, with policy iteration used to solve for Q for each draw of r. This makes BIRL method computationally intensive for large state spaces similar to NFXP. Overall BIRL is similar to the Bayesian MCMC methods discussed in section 3, except that those methods use stochastic updates of Q similar to those produced by Q-learning to gradually compute Q over the course of the MCMC simulations rather than use policy iteration to find Q for each MCMC draw of r.

A seminal contribution to the IRL literature is Ziebart et al. (2008), who introduced *maximum entropy IRL* as a means of justifying the use of the softmax/logit probability for $\pi(a|s)$ as the probability with maximum entropy subject to certain constraints that make expectations of certain functions known as *features* equal the sample averages of these features.[16] They proposed to estimate the parameters of a reward function that is a linear combination of a set of known features using a partial likelihood for observed (a, s) pairs that is identical to the partial likelihood (24) used in the DDC literature.[17] They applied this method to model the route choices of 25 Yellow cab drivers in Pittsburgh with a reward function that involves four features: road type, speed, lanes and transitions. They showed that the estimated model could correctly predict nearly 80% of the cab paths in a holdout or evaluation sample and that their new method was a significant improvement over other IRL methods such as max-margin.

These ideas evolved into the already cited work of Haarnoja et al. (2017) who characterized π as the policy that maximizes the expected discounted entropy-regularized payoff given in equation (9). Various criteria are used to estimate r such as "occupancy matching" (Ho and Ermon (2016), Yue et al. (2023)). For our purposes we focus on IRL where the likelihood function is the estimation criterion, such as Ermon et al. (2015), Zeng et al. (2023), Zeng et al. (2024), Barzegary and Yoganarasimhan (2022) and Kang et al. (2025).[18] This results in a nearly complete equivalence between the DDC and IRL literatures, with identical softmax/logit formulas for the policy $\pi(a|s)$ (16) that depend on the choice-specific or "soft Q" values that satisfy the "soft Bellman equation" in equation (13). Overall, DDC and maximum entropy IRL result in mathematically identical optimization problems for recovering r from data on states and actions of a sample of DMs. The main difference is in the algorithms used to solve the DM's DP problem.

In DDC the Q functions are computed by non-stochastic algorithms such as SA or PI, or in large state spaces by parametrizing Q or the value function V and finding parameters ϕ that minimize the squared "Bellman residuals" as in the SEES estimator, (26). All of these methods can be described as *model-based* because they require knowledge of the transition probability $p(s'|s, a)$, or at least p must be estimable from data. Adams et al. (2022) notes that most IRL methods are model-based. However, we will focus on model-free IRL that does not require specification of a function form for p or even require that it be explicitly estimated non-parametrically. Instead, new offline IRLs inspired model-free methods such as Q-learning are able to approximate the Q functions using only

[16] Ziebart et al. (2010) extended these ideas to MDPs and called it *Maximum causal entropy RL*.

[17] The full likelihood (21) is not frequently used as the estimation criterion in IRL because of the model-free or non-parametric approach that bypasses the need to explicity specify or estimate the transition probability $p(s'|s, a)$.

[18] Zeng et al. (2023) and Zeng et al. (2024) maximize a discounted log-likelihood function.

realizations from p without knowing p itself. But how is it possible to generate realizations s' from $p(s'|s,a)$ without knowing p itself?

Model-free IRL solves this problem in a clever way: it uses already realized transitions in demonstration datasets to provide the "simulator" needed by RL algorithms to train the Q and values functions. The insight is that the data on observed states and decisions provides the "environment" and the state transitions needed by RL to find the value and Q functions needed to recover r, reminiscent of the idea of *experience replay* in the deep Q learning literature, Mnih et al. (2015).[19] To get a better understanding of these methods, we will provide a more in-depth discussion of two recent contributions the IRL literature that can recover r from data in a model-free manner. One can be regarded as an IRL version of the CCP estimator and the other is similar to the SEES estimator (26). Both of these methods use parametric function approximation methods (expressing V and Q as outputs of neural networks or linear combinations of a set of basis functions) that enables them to handle problems with large state spaces.

Adusumilli and Eckhardt (2025) (AE) introduce a model-free two step CCP estimator similar the one described in the previous section but using RL-inspired methods to avoid estimating or making parametric assumptions about $p(s'|s,a)$. They do assume that the reward function is a linear combination of known "features", i.e. $r_\theta(s,a) = \sum_{k=1}^{K} \theta_k r_k(s,a)$. They use an method from RL, *temporal difference* (TD) to approximate the functions R and Q_ϵ (defined in equations (20) and (22), respectively) that enter as covariates in the CCP π in equation (23), using transitions in the data to estimate these functions without having to explicitly estimate $p(s'|s,a)$. To accommodte empirical applications with large state spaces they propose parameterizing R and Q_ϵ as linear combinations of a set of known basis functions that depend on (s,a). Let $\nu(s,a) = \{\nu_1(s,a),\dots,\nu_J(s,a)\}$ denote J basis functions that are used to approximate R nad Q_ε by a regression that uses the basis functions evaluated at values of (s,a) in the data set as regressors. We will describe AE's approach to approximating R, since the same procedure is used to approximate Q_ε and is omitted for brevity. Let R_ϕ denote an approximation to R based on a $J \times 1$ parameter vector ϕ of the form $R_\phi(s,a) = \nu(s,a)\phi = \sum_{j=1}^{J} \theta_j \nu_j(s,a)$. AE propose to to choose $\hat{\phi}$ to minimize the mean-squared TD error given by

$$\hat{\phi} = \underset{\phi}{argmin}\, E\left\{[\vec{r}(s,a) + \beta R_\phi(s',a') - R_\phi(s,a)]^2\right\}, \tag{28}$$

where E denotes the expectation operator over (s,a,s',a'). Using results from Tsitsiklis and Roy (1997) on the convergence of TD learning with linear function approximation, AE show that $\hat{\phi}$ can

[19] Also related is the distinction between online and offline RL. IRL generally uses offline RL since the estimation of rewards is based on already collected datasets rather than being done in real time. See the survey of offline RL by Levine et al. (2020) who note "Offline reinforcement learning algorithms hold tremendous promise for making it possible to turn large datasets into powerful decision making engines.". Yue et al. (2023) note that offline IRL also focuses on "learning from a previously collected dataset without online interaction with the environment" and "holds tremendous promise for safety-sensitive applications where manually identifying an appropriate reward is difficult but historical datasets of human demonstrations are readily available (e.g., in healthcare, autonomous driving, robotics, etc.)."

be computed using the sample expectation operator E_N in (28) using the formula

$$\hat{\phi} = \left[E_N\{\nu(s,a)[\nu(s,a) - \beta\nu(s',a')]^T\}\right]^{-1} E_N\{\nu(s,a)\vec{r}(s,a)\}, \tag{29}$$

and the implied estimate $\hat{R}(s,a) = \phi(s,a)\hat{\phi}$ of the features or "covariates" for the θ in (23). We also have to approximate the second "correction term" $\hat{Q}_\varepsilon(s,a)$, which can be estimated via regression similar to (29) but it requires a first stage non-parametric estimate of $\pi(a|s)$ to construct estimates of the expected choice-specific unobserved shocks $E\{\varepsilon(a)|s,a\} = -\log(\pi(a|s))$.[20] Using the estimated $\hat{R}(s,a)$ and $\hat{Q}_\varepsilon(s,a)$ values, they can compute a second state pseudo-maximum likelihood (PMLE) estimate of θ by maximizing the partial likelihood (24) using the formula for $\pi_\theta(a|s)$ in (23). Consistency of this first stage estimator requires specification of a sieve with J basis functions $\{\nu_1,\ldots,\nu_J\}$ that increases with sample size N at an appropriate rate and with sufficient linear independence to allow any function R to be perfectly approximated by its projection $\Pi_\nu R$ in the limit as $J \to \infty$. The PMLE is affected by "estimation noise" in the $\hat{R}$ and $\hat{Q}_\varepsilon$ functions that can introduce bias and noise. They propose an additional step that uses the PMLE to create a modified score (i.e. gradient) of the partial log-likelihood (24) to create a robust (Neyman orthogonal) version of the score function which they used to their preferred estimator $\hat{\theta}$ as the value that sets the modified score to zero following the approach of Chernozhukov et al. (2022).

AE proposed a different estimator they call AVI (for Approximate Value Iteration) that allows for nonlinear parametrizations of the R and Q_ε functions which are computed using an iterative procedure for updating the parameter $\hat{\phi}_k$ is the estimated parameters determining $\hat{R}_k(s,a) = R_{\hat{\phi}_k}(s,a)$ as follows

$$\hat{\phi}_{k+1} = \underset{\phi}{argmin}\, E_N\left\{\left[\vec{r}(s,a) + \beta R_{\hat{\phi}_k}(s',a') - R_\phi(s,a)\right]^2\right\}. \tag{30}$$

These iterations continue until the sequence $\{\hat{\phi}_k\}$ converges and the resulting estimator of R is $\hat{R} = R_{\hat{\phi}_k}$ where $\hat{\phi}_k$ is the last value of the iterative procedure (30), and similarly for $\hat{Q}_\varepsilon$.[21] AE conclude that "A range of Monte Carlo simulations using a dynamic firm entry problem, a dynamic firm entry game and two versions of the famous Rust (1987) engine replacement problem show that the proposed algorithms work well in practice."[22]

Kang et al. (2025) developed an estimator they call GLADIUS (Gradient-based Learning with

[20] AE normalize σ to 1 since it cannot be separately identified from the θ parameters.

[21] It is possible to extend their approach to reward functions $r_\theta(s,a)$ where θ does not enter as a linear combination of a set of known features $\vec{r}(s,a)$. However in that case, we can no longer express $Q_r(s,a) = R(s,a) * \theta$, and in general θ will enter Q_r in a non-linear fashion. The TD estimator can still be adapted to work in this situation except that now there must be a nested TD subroutine that recomputes $\hat{Q}_r$ as a function of θ each time θ is updated in an outer optimization algorithm that searches over θ to maximize the partial likelihood function (24).

[22] See Khwaja and Srivastava (2025) for an alternative approach to AE's TD methods that they call RLTD-CCS that combines the TD algorithm from RL with the Conditional Choice Simulator (CCS) estimator of Hotz et al. (1994) that relies on forward simulations of paths to produce a Monte Carlo estimate the $Q_\theta(s,a)$ and hence is a model-based approach to estimation.

Ascent-Descent for Inverse Utility learning from Samples) that is 1) model-free, 2) does not require a first stage non-parametric estimation of $\pi(a|s)$, and 3) does not require a parametric specification for rewards. Instead, GLADIUS is an iterative algorithm that uses gradient ascent/descent to maximize a penalized likelihood function similar to the SEES estimator (26). However instead of parametrizing r as a function of "structural parameters" θ and using using auxilliary parameters ϕ to approximate the Q functions, GLADIUS relies on flexible parametrization of the Q function as $Q_{\theta_1}(s,a)$ and the conditional expectation of the value function $E\{V(s')|s,a\}$ as $EV_{\theta_2}(s,a)$ for parameters θ_1 and θ_2 of flexible approximations to Q and EV such as DNNs. Once GLADIUS converges, returning parameters $\hat{\theta}_1$ and $\hat{\theta}_2$ rewards are estimated as the function

$$r(s,a) = Q_{\hat{\theta}_1}(s,a) - \beta EV_{\hat{\theta}_2}(s,a), \tag{31}$$

i.e. they are "backed out" from the fixed point equation for smoothed Q function, (13).

Both GLADIUS and SEES use an objective function that maximizes a penalized likelihood where the penalty term is a "mean squared Bellman error" for the Q functions. That is, if we express Q as a fixed point $Q = \Lambda_\sigma(Q)$, Q minimizes the quantity $\rho_{BE}(Q) = E\{[Q(s,a) - \Lambda_\sigma(Q)(s,a)]^2\}$ over a joint distribution of (s,a) pairs. SEES is a model-based approach that uses an assumed parametric functional form for rewards, $r_\theta(s,a)$ and knowledge (or estimated version thereof) of the transition probability' $p(s'|s,a)$ to compute the Bellman operator $\Lambda_{\theta,p,\sigma}$, where we subscript Λ_σ to emphasize its dependence on θ as well as p. Thus, SEES calculates the mean squared Bellman error $\rho_{BE}(Q)$ numerically using deterministic methods such as numerical quadrature to compute the conditional expectations of the Q functions. such as numerical quadrature to compute the conditional expectations of the Q functions.

GLADIUS is a model-free approach inspired by Q-learning that uses transitions (s',s,a) in the data to construct an estimate of the mean squared Bellman error $\rho_{BE}(Q)$. That is, GLADIUS seeks to approximate the quantity

$$\Lambda_\sigma(Q)(s,a) = r(s,a) + \beta \sum_{s'} V(s')p(s'|a,d) \tag{32}$$

by averaging realizations of the form

$$\tilde{\Lambda}_\sigma(Q)(s',s,a) = r(s,a) + \beta V(s'), \tag{33}$$

using observed (s,a,s') transitions in the sample. It follows that $E\{\tilde{\Lambda}_\sigma(s',s,a)|s,a\} = \Lambda_\sigma(Q)(s,a)$ so we can use transitions (s',s,a) in the data to calculate $\Lambda_\sigma(Q)(s',s,a)$ values, and then use these as dependent variables in a regression to estimate its conditional expectation $\Lambda_\sigma(s,a)$. This suggests a strategy of estimating Q by searching for θ_1 to minimize the mean squared temporal difference error $\rho_{TD}(Q)$ given by

$$\rho_{TD}(Q) = E\{[\Lambda_\sigma(Q)(s', s, a) - Q(s, a)]^2\}, \tag{34}$$

where, using the terminology of the RL literature, $\tilde{\Lambda}_\sigma(Q)(s', s, a) - Q(s, a)$ is the TD error.

However Kang et al. (2025) note an important complication: the true fixed point Q will minimize the mean squared Bellman error $\rho_{BE}(Q)$ but it will not necessarily minimize the mean squared TD error. To see, this, the note the following identity

$$\rho_{TD}(Q) = \rho_{BE}(Q) + \beta^2 E\{[V(s') - EV(s, a)]^2\}, \tag{35}$$

where the second term reflects expected "heteroscedasicity" in the TD error, i.e. it is the average of the conditional variances given by $H(V)(s, a) = E\{[V(s') - EV(s, a)]^2|s, a\}$. Since V is a function of Q from (14), we write V as $V(Q)$ and the heteroscedasticity term as $H(Q)$. It it follows from (35) that the Q that minimizes $\rho_{TD}(Q)$ will not necessarily be the same as the Q that minimizes $\rho_{BE}(Q)$. To deal with this issue, they express the mean squared Bellman error as $\rho_{BE}(Q) = \rho_{TD}(Q) - H(Q)$. Note that $H(Q)$ is also an unknown quantity they note that it can be approximated as the solution to

$$H(Q)(s, a) = \min_{v \in R} \sum_{s'} [V(Q)(s') - v]^2 p(s'|s, a). \tag{36}$$

Of course the v that minimizes (36) is $EV(s, a)$. Kang et al. (2025) parametrize Q and EV by DNNs as Q_{θ_1} and EV_{θ_2}, respectively, to obtain an estimation criterion that is a penalized likelihood function similar to SEES, except it is a max-min problem given by

$$(\hat{\theta}_1, \hat{\theta}_2) = \underset{\theta_1}{argmax}\ \underset{\theta_2}{argmin} \log(L(\theta_1)) - \lambda\rho(\theta_1, \theta_2), \tag{37}$$

where λ is a penalization parameter, and $\rho(\theta_1, \theta_2)$ is the estimate of the mean squared Bellman error given by

$$\rho(\theta_1, \theta_2) = \frac{1}{N}\sum_{i}^{N}[\Lambda_\sigma(Q_{\theta_1}(s_i', s_i, a_i) - Q_{\theta_1}(s_i, a_i)]^2 - \beta^2[V(Q_{\theta_1})(s_i') - EV_{\theta_2}(s_i, a_i)]^2 \tag{38}$$

Computation of the penalty term can be further reduced to limiting the summation to a subset of states s_i where the corresponding action a_i equals the *normalizing action.* The reason is that Kang et al. (2025) impose the identifying assumption that for each s there is an action a_s called the normalizing action for which $r(s, a_s)$ is known. From our discussion of identification in section 3.2, this restriction on rewards implies that we can determine all Q values by enforcing the fixed point constraint $Q = \Lambda_\sigma(Q)$ only for the subset of pair (s, a_s) which enables us to determine the corresponding Q at the normalizing action, $Q(s, a_s)$ for all $s \in S$. Then given $Q(s, a_s)$ the remaining $Q(s, a)$ values are determined from $\pi(a|s)$ using the Hotz-Miller inversion formula (27).

Kang et al. (2025) establish an important result on the global convergence of GLADIUS. First, under a "realizability assumption" that there exists θ_1^* such that $Q^* = Q^*_{\theta_1^*}$, where Q^* is the true Q function (as well as some other regularity conditions), they show that if $\hat{Q}_{T,N}$ is the estimate of Q^* after T iterations of the GLADIUS algorithm using a data set with N observations on agent's states and transitions (s', s, a), then

$$E_N \left\{ [Q^*(s,a) - \hat{Q}_{T,N}(s,a)]^2 \right\} \leq O(1/T) + O(1/N). \tag{39}$$

They note that "To the best of our knowledge, no prior work has proposed an algorithm that guarantees global optimum convergence of the minimization problem that involves the [mean squared Bellman residual term]". They compare GLADIUS to several other DDC estimation methods using the Rust (1987) bus engine problem and a high dimensional version of this problem where they added between 2 and 100 additional irrelevant state variables (i.e. variables that have no effect on cost of maintaining the bus or its mileage transitions, and thus on the Q values and decision rule π) that take on 20 possible values each, thus massively increasing the size of the state space. They conclude that "We find that, on average, our method performs quite well in recovering rewards in both low and high-dimensional settings. Further, it has better/on-par performance compared to other benchmark algorithms in this area (including algorithms that assume the parametric form of the reward function and knowledge of state transition probabilities) and is able to recover rewards even in settings where other algorithms are not viable."[23]

We have provided a very selective survey of the IRL literature, going into depth on two recent contributions to this literature that are particularly closely related to and we think particular promising new methods for structural estimation of DDC models. We have emphasized the distinction between model-based and model-free estimation methods, but we wish to make clear that not all IRL methods are model-free. For example Zeng et al. (2024) use non-parametric methods to estimates p to provide a "generative world model" in a model-based version of IRL and "Through extensive experiments, we demonstrate that our algorithm outperforms existing benchmarks for offline IRL and Imitation Learning, especially on high-dimensional robotics control tasks." Thus, it is not obvious that model-free approaches are necessarily superior to model-based approaches, though we agree that good models of p should be used, since "inaccurate models of the world obtained from finite data with limited coverage could compound inaccuracy in estimated rewards." It is also clear that if we are interested in doing a counterfactual that involves changes p model-based methods will probably be necessary to estimate/construct this new p and then train the model to calculate the new optimal decision rule $\pi(a|s)$ corresponding to counterfactual p.

We also do not want to leave an impression that the two studies we have focused on are the first to use function approximation methods such as DNNs to estimate models with large scale state

[23] Kang et al. (2025) also note GLADIUS corrects a problem in IRL methods that use "occupancy matching" as the estimation criterion such as Ho and Ermon (2016), Garg et al. (2022), and Yue et al. (2023). They show that occupancy matching is not guaranteed to consistency estimate the true Q function, and "This implies that r cannot be inferred from Q using the Bellman equation after deriving Q using occupancy matching."

spaces in the IRL literature. As the survey by Gleave and Toyer (2022) notes: “algorithms based on MCE IRL have scaled to high-dimensional environments. Maximum Entropy Deep IRL Wulfmeier et al. (2016) was one of the first extensions, and is able to learn rewards in gridworlds from pixel observations. More recently, Guided Cost Learning Abbeel (2016) and Adversarial IRL Fu et al. (2018) have scaled to MuJoCo continuous control tasks. Given its popularity and accomplishments we focus on MCE IRL in the remainder of this document; we refer the reader to Jeon et al. (2020) for a broader overview of reward learning.” For example, in the introduction we noted two large scale studies using IRL: the study by Barnes et al. (2024) that uses a version of IRL called Receding Inverse Horizon Planning to infer preferences over route choice using data from Google Maps, and the study by Zhao and Liang (2023) who used AIRL and showed that it outperforms competing methods in an application to inferring context-dependent preferences for route choice using data on taxi drivers in Shanghai. However all this work indicates the promise of IRL methods for estimation of large scale DDC problems.

5 Conclusion

This selective survey has drawn connections between the literature on structural estimation of dynamic discrete choice (DDC) models and the literature on inverse reinforcement learning (IRL), and their common ancestor: Markovian Decision Processes (MDP). MDPs provide a recipe for how to model the behavior of an intelligent agent who makes sequential decisions to maximize the expected, discounted value of a stream of rewards. Though MDPs can be viewed as specialized in some respects and some of their assumptions (expected utility, discounted utility) are inconsistent with empirical evidence on human decision making, it is a powerful way to approximate intelligent behavior that has opened up a huge range of empirical applications.

Despite the common ancestry, there are clear differences in the practical reasons for doing DDC and IRL modeling. In economics, DDC and other types of structural models (i.e. dynamic, stochastic general equilibrium models in macroeconomics) are used to provide counterfactual simulations to assess the impact on economics outcomes and welfare. Structural models overcome a limitation of reduced form models, which treat policy variables as a covariate and rely on historical variation in policies to make counterfactual predictions. Structural models are able to produce counterfactual predictions of the impact of novel policies that have never been enacted or even considered previously. This is why structural models are increasingly used for policymaking because it is cheaper and faster to “crash test” new policy ideas via counterfactual simulations in an “offline” environment rather than trying out new policies “online” and learning by trial and error.

In the AI and ML literatures, IRL has proven useful as a way to supply a reward function used by Reinforcement Learning (RL) algorithms for offline training of intelligent algorithms in robotics and AI. There is a wide range of complex, hard to formalize tasks where the appropriate reward function is unclear *a priori*. IRL is able to infer this reward function from limited human “demonstration data.” For example, Christiano et al. (2017) notes that “For sophisticated reinforcement learning

(RL) systems to interact usefully with real-world environments, we need to communicate complex goals to these systems. In this work, we explore goals defined in terms of (non-expert) human preferences between pairs of trajectory segments. We show that this approach can effectively solve complex RL tasks without access to the reward function, including Atari games and simulated robot locomotion, while providing feedback on less than 1% of our agent's interactions with the environment. This reduces the cost of human oversight far enough that it can be practically applied to state-of-the-art RL systems. To demonstrate the flexibility of our approach, we show that we can successfully train complex novel behaviors with about an hour of human time. These behaviors and environments are considerably more complex than any which have been previously learned from human feedback."

Another important difference in the IRL and DDC literatures is that the latter is more focused on statistical inference of the estimated reward function and deriving approximate distributions that reflect how estimates are affected by sampling errors. As a result, many DDC applications involve estimation of smaller, more parsimonious models with the goal of using them to gain new insights into human decision making and test behavioral hypotheses. The IRL literature is more focused on uncovering a reward function that can be used to train large scale applications using RL. IRL has been used to uncover reward functions for concrete problems such as route choice that can require estimation of thousands and sometimes even millions of parameters. The large number of parameters and complexity of such reward functions makes it challenging to evaluate them from the perspective of statistical inference and insight into human behavior. However the "proof is in the pudding" — the highly sophisticated and intelligent behavior in robotic and AI applications that have been learned after extensive RL training using the reward functions provided by IRL (which are in turn learned from more limited observations of human behavior).

The DDC and IRL literatures have both greatly benefitted from the use of Deep Neural Networks (DNNs) to approximate value functions in problems with large state spaces. Though it is still not formally established, the use of DNNs and other function approximation methods do seem to break Bellman's "curse of dimensionality" allowing us to formulate and estimate increasingly large and more realistic models.

A challenging area for further work is to relax and test the strong rationality assumptions underlying DDC and IRL models. There are two key assumptions: 1) rational expectations (i.e. the subjective beliefs about the decision-dependent law of motion for state variables coincides with the true one), and 2) the ability to perfectly optimize given one's beliefs. If these assumptions do not hold, rewards inferred from human behavior will be distorted by our attempt to "rationalize" behavior that may not be fully rational. The notion that human behavior is more likely to be only "boundedly rational" goes back to the work of Simon (1957), and it is exemplified by the fact that chess algorithms trained using RL such as Alpha-Zero consistently beat the best human Grand Masters. RL was able to do this without any need for human data and IRL because 1) the reward function is known *a priori* (i.e. 1 if win, 0 if lose), and 2) the algorithm as trained via self-play.

Self-play is an example of the “model-free” approach used in IRL and RL that avoids taking a stand on the “law of motion” of the state variables. Though estimation using fewer assumptions is generally an attribute, it does rely on the implicit assumption that agents we are modeling have rational expectations. To the extent that humans are boundedly rational, we would like to understand whether suboptimal behavior is due to distorted or irrational beliefs or the inability to perfectly optimize. In simpler environments that are less strategically complex than chess, humans may be more likely to perfectly optimize but if their beliefs are not rational, the resulting behavior will still be suboptimal from the standpoint of a rational observer.

Anderson et al. (2025) showed that elite tennis professionals use suboptimal serve strategies, and demonstrated how their behavior can be explained by distorted subjective beliefs about their own strengths and weaknesses as servers as well as their opponents. This opens a path for DDC and IRL models to improve human performance using data and model-free approaches that enables them to correct distorted subjective beliefs. However there are many high stake strategic situations (e.g. a wartime scenario) where the relevant data may be too scarce, leaving any decision maker no choice but to rely on an implicit “mental model” of the situation. In such situations neither DDC nor IRL methods may prove useful because of their dependence on good models of beliefs. More work needs to be done on how it might possible to construct relatively accurate mental models (i.e. beliefs about how different actions affect outcomes), in unique situations where relevant data is extremely limited.

We conclude that the limitations confronting DDC and IRL are not as much computational but rather inherent limitations on our ability to infer underlying rewards and beliefs from data on states and actions as we noted in our discussion of the identification problem in section 2.4. The solution to the identification problem requires us to impose strong prior restrictions (assumptions) about rewards and beliefs. If these assumptions are wrong, counterfactual predictions from these models may not be accruate even though the models provide a good fit to the demonstration data used to estimate them. However practical experience suggests that even if our assumptions may not be precisely correct and human decision makers may not be not fully rational, the assumptions used to identify DDC and IRL models appear to be sufficiently good approximations to produce realistic counterfactual simulations and enable the training artificially intelligent agents to perform a huge variety of tasks that previously we assumed only intelligent human beings could perform.

References

Abbeel, Chelsea Finn Sergey Levine Pieter, "Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization," *arXiv*, 2016, *arXiv:1603.00448v3.*

Abbring, Jaap and Øystein Daljord, "Identifying the discount factor in dynamic discrete choice models," *Quantitative Economics*, 2020, *11* (2), 471–501.

Adams, Stephen, Tyler Cody, and Peter A. Beling, "A survey of inverse reinforcement learning," *Artificial Intelligence Review*, 2022, *55*, 4307–4436.

Adusumilli, Kaun and Dita Eckhardt, "Temporal Difference Estimation of Dynamic Discrete Choice Models," *Review of Economic Studies*, 2025, *forthcoming.*

Aguirregabiria, Victor and Pedro Mira, "Swapping the nested fixed point algorithm: A class of estimators for discrete Markov decision models," *Econometrica*, 2002, *70*, 1519–1543.

_ **and** _ , "Dynamic discrete choice structural models: A survey," *Journal of Econometrics*, 2010, *156*, 38–67.

Anderson, Axel, Jeremy Rosen, John Rust, and Kin-Ping Wong, "Disequilibrium Play in Tennis," *Journal of Political Economy*, 2025, *133* (1), 190–251.

Arora, Saurabh and Prashant Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, 2021, *297* (103500).

Azar, Nematollah Ab, Aref Shahmansoorian, and Mohsen Davoudi, "From inverse optimal control to inverse reinforcement learning: A historical review," *Annual Reviews in Control*, 2020, *50*, 119–138.

Bajari, Patrick, C. Lanier Benkard, and John Levin, "Estimating Dynamic Models of Imperfect Competition," *Econometrica*, 2007, *75* (5), 1331–1370.

Barnes, M., M.Abueg, O. F. Lange, M. Deeds, J. Trader, D. Molitor, M. Wulfmeier, and S. O'Banion, "Massively Scalable Inverse Reinforcement Learning in Google Maps," *OpenReview.net*, 2024, pp. 1–22.

Barto, A. G., S. J. Bradtke, and S. P. Singh, "Learning to Act with Real Time Dynamic Programming," *Artificial Intelligence*, 1995, *72*, 81–131.

Barzegary, Ebrahim and Hema Yoganarasimhan, "A recursive partitioning approach for dynamic discrete choice modeling in high dimensional settings," *arXiv*, 2022, *arXiv:2208.01476.*

Bayer, Federico A.Bugni Peter Arcidiacono Patrick and Jonathan James, "Approximating High-Dimensional Dynamic Models: Sieve Value Function Iteration," in Eugene Choo and Matthew Shum, eds., *Structural Econometric Models*, Emerald Publishing Ltd., 2013, pp. 44–74.

Bellman, Richard, *Dynamic Programming*, 4th ed., Princeton University Press, 2017.

_ **and Stuart Dreyfus**, "Functional Approximations and Dynamic Programming," *Mathematical Tables and Other Aids to Computation*, 1959, *13* (68), 247–251.

Benitez-Silva, Hugo, George Hall, Hitsch Günter J, Giorgio Pauletto, and John Rust, "A comparison of discrete and parametric approximation methods for continuous-state dynamic programming problems," *manuscript, Yale University*, 2000.

Bertsekas, Dimitri P., *Dynamic Programming and Optimal Control (volumes 1 and 2)*, 4th ed., Athena Scientific, Belmont Massachusetts, 2017.

Chernozhukov, Victor, Juan Carlos Escanciano, Hidehiko Ichimura, Whitney K. Newey, and James M. Robins, "Locally Robust Semiparametric Estimation," *Econometrica*, 2022, *90* (4), 1051–1535.

Christiano, Paul F., Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei, "Deep Reinforcement Learning from Human Preferences," *31st Conference on Neural Information Processing Systems*, 2017.

Dearing, Adam and Jason Blevins, "Efficient and Convergent Sequential Pseudo-Likelihood Estimation of Dynamic Discrete Choice Games," *Review of Economic Studies*, 2025, *92*, 981–1021.

Ermon, Stefano, Yexiang Xue, Russell Toth, Bistra Dilkina, Richard Bernstein, Theodoros Damoulas, Patrick Clark, Steve DeGloria, Andrew Mude, Christopher Barrett, and Carla P. Gomes, "Learning large-scale dynamic discrete choice models of spatio-temporal preferences with application to migratory pastoralism in East Africa," *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, p. 644–650.

Frederick, Shane, George Loewenstein, and Ted O'Donoghue, "Time Discounting and Time Preference: A Critical Review," *Journal of Economic Literature*, 2002, *XL*, 351–401.

Fu, Justin, Katie Luo, and Sergey Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *Proceedings of ICLR*, 2018, pp. 1—-15.

Garg, Divyansh, Shuvam Chakraborty, Chris Cundy, Jiaming Song, Matthieu Geist, and Stefano Ermon, "IQ-Learn: Inverse soft-Q Learning for Imitation," *arXiv*, 2022, *arXiv:2106.12142v4.*

Gillingham, Kenneth, Fedor Iskhakov, Anders Munk-Nielsen, John Rust, and Bertel Schjerning, "Equilibrium Trade in Automobiles," *Journal of Political Economy*, 2022, *130* (10), 2534–2593.

Gleave, Adam and Sam Toyer, "A Primer on Maximum Causal Entropy Inverse Reinforcement Learning," *arXiv*, 2022, *arXiv:2203.11409v1.*

Haarnoja, Tuomas, Haoran Tang, Pieter Abbeel, and Sergey Levine, "Reinforcement Learning with Deep Energy-Based Policies," *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Ho, Jonathan and Stefano Ermon, "Learning large-scale dynamic discrete choice models of spatio-temporal preferences with application to migratory pastoralism in East Africa," *Advances in neural information processing systems*, 2016, *29.*

Hotz, Joseph V. and Robert A. Miller, "Conditional choice probabilities and the estimation of dynamic models," *Review of Economic Studies*, 1993, *60*, 497–529.

_ , _ , Seth Sanders, and Jeffrey Smith, “A simulation estimator for dynamic models of discrete choice,” *Review of Economic Studies*, 1994, *61*, 265–289.

Howard, Ronald A., *Dynamic Programming and Markov Processes*, Technology Press, Cambridge, MA, 1960.

Imai, Susumu, Neelam Jain, and Andrew Ching, “Bayesian Estimation of Dynamic Discrete Choice Models,” *Econometrica*, 2009, *77* (6), 1865–1899.

Iskhakov, Fedor, Jinhyuk Lee, John Rust, Bertel Schjerning, and Kyoungwon Seo, “Constrained Optimization Approaches to Estimation of Structural Models: Comment,” *Econometrica*, 2016, *84* (1), 365–370.

_ , John Rust, and Bertel Schjerning, “Recursive Lexicographical Search: Finding All Markov Perfect Equilibria of Finite State Directional Dynamic Games,” *Review of Economic Studies*, 2015, *83* (2), 658–703.

Jeon, Hong Jun, Smitha Milli, and Anca Dragan, “Reward-rational (implicit) choice: A unifying formalism for reward learning,” in “34th Conference on Neural Information Processing Systems” 2020.

Jeon, Wonseok, Chen-Yang Su, Paul Barde, Thang Doan, Derek Nowrouzezahrai, and Joelle Pineau, “Regularized Inverse Reinforcement Learning,” *OpenReview.net*, 2021, *ICLR2021.*

Jiang, Nan and Tengyang Xie, “Offline Reinforcement Learning in Large State Spaces: Algorithms and Guarantees,” *manuscript, University of Illinois*, 2024.

Kalman, R. E., “When is a linear control system optimal?,” *Journal of Basic Engineering*, 1964, *86*, 119–138.

Kalouptsidi, Myrto, Paul T. Scott, and Eduardo Souza-Rodrigues, “Identification of counterfactuals in dynamic discrete choice models,” *Quantitative Economics*, 2021, *12* (2), 351–403.

Kang, Enoch H., Hema Yoganarasimhan, and Lalit Jain, “An Empirical Risk Minimization Approach for Offline Inverse RL and Dynamic Discrete Choice Model,” *arXiv*, 2025, *arXiv:2502.14131.*

Khwaja, Ahmed and Sonal Srivastava, “Reinforcement Learning Based Computationally Faster Conditional Choice Simulation Estimation of Dynamic Discrete Choice Models,” *manuscript, Judge Business School, University of Cambridge*, 2025.

Kristensen, Dennis, Patrick K. Mogensen, Jong Myun Moon, and Bertel Schjerning, “Solving dynamic discrete choice models using smoothing and sieve methods,” *Journal of Econometrics*, 2021, *223*, 328–360.

Levine, Sergey, Aviral Kumar, George Tucker, and Justin Fu, “Offline reinforcement learning algorithms hold tremendous promise for making it possible to turn large datasets into powerful decision making engines,” *arXiv*, 2020, *arXiv:2005.01643v3.*

Lucas, Jr. Robert E., “Econometric Policy Evaluation: A Critique,” in Alan Meltzer, ed., *The Phillips Curve and Labor Markets. Carnegie-Rochester Conference Series on Public Policy. Vol. 1*, Elsevier, 1976, pp. 19–46.

Luo, Yao and Peijun Sang, “Efficient Estimation of Structural Models via Sieves,” *arXiv*, 2024, *arXiv:2204.13488v2.*

Machina, Mark J., “Choice Under Uncertainty: Problems Solved and Unsolved,” *Economic Perspectives*, 1987, *101*, 121–154.

Magnac, Thierry and David Thesmar, “Identifying Dynamic Discrete Decision Processes,” *Econometrica*, 2002, *70*, 801–816.

Maskin, Eric and Jean Triole, “Markov Perfect Equilibrium: I. Observable Actions,” *Journal of Economic Theory*, 2001, *100*, 191–219.

McFadden, Daniel L., “Conditional Logit Analysis of Qualitative Choice Behavior,” in P. Zarembka, ed., *Frontiers in Econometrics*, Academic Press, 1973, pp. 105–142.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature*, 2015, *518*, 518–533.

Mombaur, Katja, Anh Truong, and Jean-Paul Laumond, “From human to humanoid locomotion—an inverse optimal control approach,” *Autonomous Robotics*, 2010, *28*, 369–383.

Muth, John F., “Rational Expectations and the Theory of Price Movements,” *Econometrica*, 1961, *29* (3), 315–335.

Ng, Andrew Y. and Stuart Russell, “Algorithms for Inverse Reinforcement Learning,” *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000, p. 663–670.

Norets, Andriy, “Inference in Dynamic Discrete Choice Models With Serially orrelated Unobserved State Variables,” *Econometrica*, 2009, *77* (5), 1665–1682.

Osa, T., J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, “An Algorithmic Perspective on Imitation Learning,” *Foundations and Trends in Robotics*, 2018, *7*, 1–179.

Puterman, Martin L. and Shelby L. Brumelle, “On the Convergence of Policy Iteration in Stationary Dynamic Programming,” *Mathematics of Operations Research*, 1979, *4*, 60–69.

Ramachandran, Deepak and Eyal Amir, “Bayesian Inverse Reinforcement Learning,” *Proceedings IJCAI*, 2007, p. 2586–2591.

Russell, Stuart J. and Peter Norvig, *Artificial Intelligence A Modern Approach*, 4th ed., Pearson, 2022.

Rust, John, “Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher,” *Econometrica*, 1987, *55* (5), 993–1033.

_ , "Maximum Likelihood Estimation of Discrete Control Processes," *SIAM Journal on Control and Optimization*, 1988, *26* (5), 1006–1024.

_ , "Structural Estimation of Markov Decision Processes," in R. Engle and D. McFadden, eds., *The Handbook of Econometrics, Volume 4*, Elsevier, 1994, pp. 3082–3139.

Samuelson, Paul, "A Note on Measurement of Utility," *Review of Economic Studies*, 1937, pp. 155–161.

Sargent, Thomas J., "Estimation of Dynamic Labor Demand Schedules under Rational Expectations," *Journal of Political Economy*, 1978, *86* (6), 1009–1044.

Silver, D., A. Huang, C.J. Maddison, A Guez, L Sifre, G Van Den Driessche, J. Schrittwieser, I Antonoglu, V Panneershelvam, M Lanctot, and S Dielman, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, 2016, *529*, 484–489.

Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science*, 2018, *362*, 1140–1144.

Simon, Herbert A., *Administrative Behavior: A Study of Decision-Making Processes in Administrative Organization*, 2nd ed., Macmillan, New York, 1957.

Su, Che-Lin and Kenneth L. Judd, "Constrained Optimization Approaches to Estimation of Structural Models," *Econometrica*, 2012, *80* (5), 2213–2230.

Tsitsiklis, John N., "Asynchronous Stochastic Approximation and Q-Learning," *Machine Learning*, 1994, *16*, 185–202.

_ **and Benjamin Van Roy**, "An analysis of temporal-difference learning with function approximation," *IEEE transactions on automatic control*, 1997, *42*, 674–690.

von Neumann, John and Oskar Morgenstern, *The Theory of Games and Economic Behavior*, Princeton University Press, 1944.

Wang, Siyue, Zhaorun Chen, Zhuokai Zhao, Chaoli Mao, Yiyang Zhou, Jiayu He, and Albert Sibo Hu, "ESCIRL: Evolving Self-Contrastive IRL for Trajectory Prediction in Autonomous Driving," *8th Conference on Robot Learning (CoRL 2024)*, 2025, pp. 1–20.

Watkins, C. J. C. H., *Learning from Delayed Rewards*, PhD Thesis, Kings College, Cambridge UK, 1989.

_ **and Peter Dayan**, "Technical Note: Q-Learning," *Machine Learning*, 1992, *8*, 279–282.

Wulfmeier, Markus, Peter Ondrúška, and Ingmar Posner, "Maximum Entropy Deep Inverse Reinforcement Learning," *arXiv*, 2016, *arXiv:1507.04888v3.*

Yue, Sheng, Guanbo Wang, Wei Shao, Zhaofeng Zhang, Sen Lin, Ju Ren, and Junshan Zhang, "CLARE: Conservative Model-Based Reward Learning for Offline Inverse Reinforcement Learning," in "The Eleventh International Conference on Learning Representations" 2023.

Zeng, Siliang, Chenliang Li, Alfredo Garcia, and Mingyi Hong, "When Demonstrations Meet Generative World Models: A Maximum Likelihood Framework for Offline Inverse Reinforcement Learning," *37th Conference on Neural Information Processing Systems*, 2023.

_ , Mingyi Hong, and Alfredo Garcia, "Structural Estimation of Markov Decision Processes in High-Dimensional State Space with Finite-Time Guarantees," *Operations Research*, 2024, *73* (2).

Zhao, Zhan and Yuebing Liang, "A deep inverse reinforcement learning approach to route choice modeling with context-dependent rewards," *Transportation Research Part C*, 2023, *149.*

Ziebart, Brian D., Andrew Maas, J. Andrew Bagnell, and Anind K. Dey, "Maximum Entropy Inverse Reinforcement Learning," *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008, *55* (5), 1433–1438.

_ , J. Andrew Bagnell, and Anind K. Dey, "Modeling Interaction via the Principle of Maximum Causal Entropy," *Proceedings of the 27th International Conference on Machine Learning*, 2010.