An Empirical Risk Minimization Approach for Offline Inverse RL and Dynamic Discrete Choice Model

Enoch Hyunwook Kang*[1], Hema Yoganarasimhan[1] and Lalit Jain[1 2]

Econometric Society Summer School in Dynamic Structural Econometrics 2025

1 Foster School of Business, University of Washington

2 Google Deepmind

We will discuss...

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Example: Walmart+'s optimal coupon timing

- Free delivery service if subscribed
- $12.95/month subscription, renewed monthly
- Each month, subscribers decide whether to continue/discontinue Walmart + membership
- Walmart wants to find the optimal coupon timing for subscribers
  - Give the subscriber a 50% discount this month
    - Walmart loses money now but may make more money in the future
  - Don't give subscriber a 50% discount this month.
    - Walmart doesn't lose money now, customer may churn

Example: Walmart+'s optimal coupon timing

state at month 0 (s_0)

action at month 0 (a_0) – Coupon

state at month 1 (s_1)

action at month 1 (a_1) – No coupon

state at month 2 (s_2)

action at month 2 (a_2) – No coupon

state at month 3 (s_3)

action at month 3 (a_3) – No coupon

Customer Churn

Page views
Session Duration
Cart addition
Abandonment Rate
Purchase Frequency
Average Order Value
Product Category Preference
Device Usage Proportion
Average Items per Cart
Search Usage Frequency
Time Between Purchases
Peak Activity time
Shipping Cost Tolerance
Average Scroll Depth
Loyalty Program Engagement
...

Example: Walmart+'s optimal coupon timing

state at month 0 (s_0)

action at month 0 (a_0) – Coupon

state at month 1 (s_1)

action at month 1 (a_1) – No coupon

state at month 2 (s_2)

action at month 2 (a_2) – No coupon

state at month 3 (s_3)

action at month 3 (a_3) – No coupon

Customer Churn

Walmart's utility?

$u(s_0, a_0)$

$u(s_1, a_1)$

$u(s_2, a_2)$

$u(s_3, a_3)$

(−) immediate subscription revenue
(+) Increased sales
(+) Enhanced customer loyalty
(+) Increased lifetime value
….

Unknown utilities

Example: Walmart+'s optimal coupon timing

Given: Dataset of $(s_0, a_0, s_1, a_1, \dots,)$

Unknown: $(u(s_0, a_0), u(s_1, a_1), \dots,)$

Goal: Infer the unknown utility function $u(s_t, a_t)$

Canonical Dynamic Discrete Choice Model problem

Question: Can we apply existing DDC methods?

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Single-agent [2] Markov Decision Process (MDP)

Given offline dataset $\mathcal{D} = (s_0, a_0, s_1, a_2, \ldots)$, where s_t = state at t, a_t = action at t

$\mathcal{D}$ is collected from agent's interaction with the Markov Decision Process $(\mathcal{S}, \mathcal{A}, P, \nu_0, r, G, \beta)$:

- $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes a finite action space
- $P \in \Delta_{\mathcal{S}}^{\mathcal{S}\times\mathcal{A}}$ is a Markovian transition kernel, $\nu_0 \in \Delta_{\mathcal{S}}$ is the initial state distribution over $\mathcal{S}$
- $\beta \in (0, 1)$ a discount factor
- $r(s, a) + \epsilon_a$ is the reward agent privately observes [1]
  - $r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is a deterministic part of reward.
  - $\epsilon_a \overset{i.i.d.}{\sim} G(\delta, 1)$ is the random part of the reward, where G is Type 1 Extreme Value (T1EV) distribution (i.e., Gumbel distribution)

[1] This reward form is often said to satisfy linear additivity and conditionally independence

[2] We will discuss extension to the unobserved heterogeneity setup at the end

Notations

Given an initial state $s_0 \sim \nu_0$, we define the distribution of state-action sequences for policy π over the sample space $(\mathcal{S} \times \mathcal{A})^\infty = \{(s_0, a_0, s_1, a_1, \ldots) : s_h \in \mathcal{S}, a_h \in \mathcal{A}, h \in \mathbb{N}\}$ as $\mathbb{P}_{\nu_0,\pi}$

We also use $\mathbb{E}_{\nu_0,\pi}$ to denote the expectation with respect to $\mathbb{P}_{\nu_0,\pi}$

We further define:

- Optimal policy $\pi^* := \text{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \mathbb{E}_{\nu_0,\pi,G} \left[\sum_{h=0}^{\infty} \beta^h (r(s_h, a_h) + \epsilon_{a_h})\right]$
- Value function $V^*(s) := \max_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \mathbb{E}_{\nu_0,\pi,G} \left[\sum_{h=0}^{\infty} \beta^h (r(s_h, a_h) + \epsilon_{a_h}) \mid s_0 = s\right]$
- Q-function $Q^*(s, a) := r(s, a) + \beta \cdot \mathbb{E}_{s' \sim P(s,a)} \left[V^*(s') \mid s, a\right]$

Dynamic Discrete Choice (DDC) Model estimation problem

Assumption 1 (Rationality)

Data $\mathcal{D}$ is collected from agent's interaction with the MDP $(\mathcal{S}, \mathcal{A}, P, \nu_0, r, G, \beta)$ while following the optimal policy π^*.

Definition 1 (Dynamic Discrete Choice (DDC) model estimation problem)

Given offline data $\mathcal{D}$, under Assumption 1, we would like to find a function $\hat{r} \in \mathcal{R} \subseteq \mathbb{R}^{\bar{\mathcal{S}} \times \mathcal{A}}$:

$$\operatorname*{argmin}_{\hat{r} \in \mathcal{R} \subseteq \mathbb{R}^{\bar{\mathcal{S}} \times \mathcal{A}}} \mathbb{E}_{(s,a) \sim \nu_0, \pi^*} [(\hat{r}(s,a) - r(s,a))^2]$$

where $\overline{\mathcal{S}} := \{s \in \mathcal{S} \mid \mathbb{P}_{\nu_0, \pi^*}(\{s_h = s\}) > 0$ for some $h \geq 0\}$ defines the optimal policy's coverage, which consists of states that are reachable with nonzero probability under the expert's optimal policy π^*. [3]

[3] Inferring rewards beyond $\bar{\mathcal{S}}$ would be ill-posed even with infinite data. Similarly, computing MSE using the π^*'s state-action distribution is natural since the goal is to recover the reward function that explains the expert's behavior.

Basic results in DDC model

Fact 1 (Optimality equations)

Under Assumption 1, we have

$$\pi^*(a \mid s) = \frac{\exp\left(Q^*(s,a)\right)}{\sum_{a' \in \mathcal{A}} \exp\left(Q^*(s,a')\right)} \quad \textit{for } a \in \mathcal{A} \tag{1}$$

$$\begin{aligned} Q^*(s,a) &= r(s,a) + \beta \cdot \mathbb{E}_{s' \sim P(s,a)}\left[V(s') \mid s, a\right] \\ &= r(s,a) + \beta \cdot \mathbb{E}_{s' \sim P(s,a)}\left[\log \sum_{a' \in \mathcal{A}} \exp(Q^*(s',a')) \mid s, a\right] + \delta + \gamma \end{aligned} \tag{2}$$

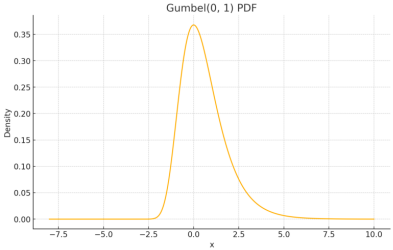
Optimality equation (1): π^* - Q^* relationship

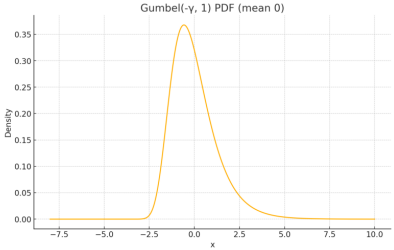
Optimality equation (2): Bellman equation with respect to Q^*

Note) T1EV (Gumbel) distribution setup

Recall that reward's random part is $\epsilon_a \overset{i.i.d.}{\sim} G(\delta, 1)$, where G is Type 1 Extreme Value (T1EV) distribution (i.e., Gumbel distribution)

- $G(0, 1)$ is not a mean 0 distribution (mean $\delta + \gamma = 0 + \gamma = \gamma$)
- $G(-\gamma, 1)$ is a mean 0 distribution (mean $\delta + \gamma = -\gamma + \gamma = 0$)





For simplicity, we will use $\epsilon_a \overset{i.i.d.}{\sim} G(-\gamma, 1)$ throughout

Note) DDC-IRL equivalence

DDC-IRL difference is about the way of explaining randomness in action choice:

- Dynamic Discrete Choice model (DDC)'s explanation:

  "There is an unobserved random state component that induces this uncertainty"

- Inverse Reinforcement Learning (IRL)'s explanation:

  "The action choice itself is random, because the agent inherently prefers a random action choice"

Fact 2

Exactly the same optimality equations in **Fact 1** *hold for DDC and IRL.*[4]

Indeed, IRL community has been rediscovering what DDC has developed. [5] [6]

[4] For details, see Appendix C of Kang et al., (2025)

[5] "Identifiability in inverse reinforcement learning", Cao et al., NeurIPS 2021

[6] "A maximum likelihood framework for offline inverse reinforcement learning" Zeng et al., NeurIPS 2023

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Recall. Optimality equations

$$\pi^*(a \mid s) = \frac{\exp\left(Q^*(s,a)\right)}{\sum_{a' \in \mathcal{A}} \exp\left(Q^*(s,a')\right)} \text{ for } a \in \mathcal{A} \tag{1}$$

$$Q^*(s,a) = r(s,a) + \beta \cdot \mathbb{E}_{s' \sim P(s,a)}\left[\log \sum_{a' \in \mathcal{A}} \exp(Q^*(s',a')) \mid s, a\right] \tag{2}$$

We would like to use them to find the r and Q^*.

For that, let's rewrite these equations.

Rewriting the optimality equation (1)

Define $\frac{\exp(Q(s,a))}{\sum_{a' \in \mathcal{A}} \exp(Q(s,a'))}$, as $\hat{p}_Q(a \mid s)$.

Then for Q, to satisfy the optimality equation (1), it must satisfy $\pi^*(a \mid s) = \hat{p}_Q(a \mid s)$.

Lemma 1

$$\left\{ Q \in \mathcal{Q} \mid \pi^*(\,\cdot \mid s) = \hat{p}_Q(\,\cdot \mid s) \quad \forall s \in \bar{\mathcal{S}} \quad a.e. \right\}$$

$$= \arg\min_{Q \in \mathcal{Q}} \; \mathbb{E}_{(s,a) \sim \pi^*, \nu_0} \left[-\log\left(\hat{p}_Q(a \mid s)\right) \right]$$

We define $\mathcal{L}_{NLL}(Q)(s,a) := -\log\left(\hat{p}_Q(a \mid s)\right)$ and call it negative log likelihood (NLL) loss.

Rewritten optimality equation (1):

$$Q^* \in \arg\min_{Q \in \mathcal{Q}} \; \mathbb{E}_{(s,a) \sim \pi^*, \nu_0} \left[\mathcal{L}_{NLL}(Q)(s,a) \right]$$

Known results for solving DDC models

Assumption 2

For all $s \in \mathcal{S}$, there exists an action $a_s \in \mathcal{A}$ set $r(s, a_s) = 0$.

Under Assumption 2, we know two ways of solving DDC models:

r, Q^* estimation

$\Updownarrow$ (Rust 1994)

1. Q^* minimizes $\mathbb{E}_{(s,a)\sim\pi^*}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]$
2. Q^* satisfies the Bellman equation $\forall s \in \overline{\mathcal{S}}, a \in \mathcal{A}$

r, Q^* estimation

$\Updownarrow$ (Hotz and Miller 1993)

1. Q^* minimizes $\mathbb{E}_{(s,a)\sim\pi^*}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]$
2. $Q^*(s,a) := \mathbb{E}_{\pi^*,\nu_0}\left[\sum_{t=0}^{\infty} \beta^t r(s_t, a_t) + \epsilon_{a_t} \mid s, a\right]$

We observe two challenges common to both approaches

1. Nestedness of optimization (No one-shot optimization)
2. Curse of dimensionality

Challenge 1: Optimization Requires Nesting

r, Q^* estimation

$\Updownarrow$ (Rust 1994)

(1) Q^* minimizes $\mathbb{E}_{(s,a)\sim\pi^*}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]$

(2) Q^* satisfies the Bellman equation $\forall s \in \overline{\mathcal{S}}, a \in \mathcal{A}$

r, Q^* estimation

$\Updownarrow$ (Hotz and Miller 1993)

(1) Q^* minimizes $\mathbb{E}_{(s,a)\sim\pi^*}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]$

(2) $Q^*(s_t, a_t) := \mathbb{E}_{\pi^*,\nu_0}\left[\sum_{t=0}^{\infty} \beta^t r(s_t, a_t)\right]$

In both cases, we need to parametrize r (to satisfy equation (2) part).

But then, we need to solve a nested optimization:

- the equation (1) part is the outer loop, which is about optimization within Q space
- the equation (2) part is the inner loop, which is about deriving Q from r

Challenge 2: Curse of dimensionality

Challenge 2: for both, solving equation (2) for a large-scale problem is challenging

- when we follow Rust (1994): we need to solve Bellman equation $Q(s,a) = r(s,a) + \beta E_{s' \sim P(s,a)} \left[\ln \left(\sum_{a \in \mathcal{A}} \exp \left(Q\left(s', a\right) \right) \right) \mid s, a \right]$
  - Computational complexity exponentially increases as the dimension increases
- when we follow Hotz and Miller (1993): we need transition function estimation to compute $Q\left(s_t, a_t\right) := \mathbb{E}_{\pi^*, \nu_0} \left[\sum_{t=0}^{\infty} \beta^t r\left(s_t, a_t\right) \right]$
  - Statistical complexity exponentially increases as the dimension increases

Walmart+'s optimal coupon timing problem

state at month 0 (s_0)

action at month 0 (a_0) – Coupon

state at month 1 (s_1)

action at month 1 (a_1) – No coupon

state at month 2 (s_2)

action at month 2 (a_2) – No coupon

state at month 3 (s_3)

action at month 3 (a_3) – No coupon

Customer Churn

Page views
Session Duration
Cart addition
Abandonment Rate
Purchase Frequency
Average Order Value
Product Category Preference
Device Usage Proportion
Average Items per Cart
Search Usage Frequency
Time Between Purchases
Peak Activity time
Shipping Cost Tolerance
...

State space size > 10^{20}

Curse of dimensionality happens!

Nonparametric function approximation to address the curse of dimensionality

There are two ways[7] to solve the Bellman equation without transition function knowledge:

- Fitted fixed point iteration (e.g., Fitted value iteration, Fitted Q-iteration)
  - John introduced this earlier
  - Studied by Adusumilli and Eckardt (2022)[8]
  - Suffers instability when combined with neural networks (next page)
- Gradient-based squared Bellman error (BE) minimization
  - No prior works (I will explain this later)
  - This paper

[7] See "Offline reinforcement learning in large state spaces: algorithms and guarantees", Jiang and Xie, Statistical Sciences (to appear). Note that importance sampling approaches are excluded as they do not guarantee the Bellman equation to hold.

[8] Adusumilli, and Eckardt, "Temporal-Difference estimation of dynamic discrete choice models", RES (to appear)

Divergence issue of Fitted fixed point iteration methods

Theorem 2 (Tsitsiklis and Roy 1996, Jiang and Xie 2025)

Fitted fixed point methods can diverge even when all of the following hold:

1. $|\mathcal{D}| = \infty$ *and the fitting is exact (i.e., perfect optimization using infinite computation).*
2. $\mathcal{F}$ *is a 1-dim linear function class that exactly captures* Q^**, i.e.,* $Q^* \in \mathcal{Q}$ *(a.k.a. realizability).*

This is not just a theoretical phenomenon[9], and instability is a huge issue in practice. [10]

This issue becomes even worse when combined with neural networks. [11]

Intuitively: 'Solving for Bellman equation involves bootstrapping unrealized functions' [12]

[9] Tsitsiklis and Roy, "Feature-based methods for large scale dynamic programming", Machine Learning (1996)

[10] Van Hasselt et al., "Deep reinforcement learning and the deadly triad", arXiv:1812.02648 (2018).

[11] Wang et al., "Instabilities of offline rl with pre-trained neural representation", ICML (2021)

[12] Jiang and Xie, "Offline reinforcement learning in large state spaces: algorithms and guarantees", Statistical Sciences (2025)

The question answered in this paper

Can a scalable gradient-based method infer rewards (or Q^* function) while provably ensuring global optimality without any assumption on reward structure/transition function knowledge?

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Terms: Bellman operators

Define the Bellman operator $\mathcal{T}$ as:

$$\mathcal{T}Q(s,a) := r(s,a) + \beta \mathbb{E}_{s' \sim P(s,a)} \left[\ln \left(\sum_{a \in \mathcal{A}} \exp\left(Q\left(s', a'\right)\right) \right) \mid s, a \right]$$

Define the empirical Bellman operator $\widehat{\mathcal{T}}$ as:

$$\widehat{\mathcal{T}}Q(s,a,s') := r(s,a) + \beta \left[\ln \left(\sum_{a \in \mathcal{A}} \exp\left(Q\left(s', a'\right)\right) \right) \mid s, a \right]$$

Note that we have $\mathbb{E}_{s' \sim P(s,a)} \left[\hat{\mathcal{T}} Q\left(s,a,s'\right) \right] = \mathcal{T}Q(s,a)$.

Term: Bellman Error (BE) loss and Temporal Difference (TD) loss

The Bellman equation:

$$Q^*(s_t, a_t) = \underbrace{r(s_t, a_t) + \beta E_{s_{t+1} \sim P(s_t, a_t)} \left[\ln \left(\sum_{a \in \mathcal{A}} \exp\left(Q^*(s_{t+1}, a)\right) \right) \mid s_t, a_t \right]}_{\mathcal{T} Q^*(s_t, a_t)}$$

We define $\mathcal{L}_{BE}(Q)(s, a) := (Q(s, a) - \mathcal{T} Q(s, a))^2$ and call it squared Bellman Error (BE) loss.

We define $\mathcal{L}_{TD}(Q)(s, a) := (Q(s, a) - \widehat{\mathcal{T}} Q(s, a, s'))^2$ and call it squared Temporal Difference (TD) loss.

Recall. Rust (1994) way

Assumption 2

For all $s \in \mathcal{S}$, there exists an action $a_s \in \mathcal{A}$ set $r(s, a_s) = 0$.

Under Assumption 2, we have:

$$r, Q^* \text{ estimation}$$

$$\Updownarrow \text{ (Rust 1994)}$$

1. Q minimizes $\mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]$
2. Q satisfies the Bellman equation $\forall s \in \overline{\mathcal{S}}, a \in \mathcal{A}$

Magnac and Thesmar (2009)

Assumption 2

For all $s \in \mathcal{S}$, there exists an action $a_s \in \mathcal{A}$ set $r(s, a_s) = 0$.

Under Assumption 2, we have:

$$r, Q^* \text{ estimation}$$

$$\Updownarrow \text{ Magnac and Thesmar (2009)}$$

1. Q minimizes $\mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]$
2. Q satisfies the Bellman equation for all $(s,\ a_s)$ with $s \in \overline{\mathcal{S}}$

Rewriting optimality equation (2)

Note that we have

$$\{Q \in \mathcal{Q} \mid Q \text{ satisfies the Bellman equation for all } (s, a_s) \text{ with } s \in \overline{\mathcal{S}}\}$$

$$= \left\{Q \in \mathcal{Q} \mid \mathcal{L}_{BE}(Q)(s, a_s) = 0 \quad \forall s \in \overline{\mathcal{S}}\right\}$$

$$= \arg\min_{Q \in \mathcal{Q}} \mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathbf{1}_{a=a_s}\mathcal{L}_{BE}(Q)(s, a)\right]$$

Magnac and Thesmar (2009) (Modified)

Assumption 2

For all $s \in \mathcal{S}$, there exists an action $a_s \in \mathcal{A}$ set $r(s, a_s) = 0$.

Under Assumption 2, we have:

$$Q = Q^*, r = r^*$$

$$\Updownarrow \text{ Magnac and Thesmar (2009)}$$

1. Q minimizes $\underbrace{\mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathcal{L}_{NLL}(Q)(s,a)\right]}_{\text{No } r \text{ related term}}$
2. Q minimizes $\underbrace{\mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathbf{1}_{a=a_s}\mathcal{L}_{BE}(Q)(s,a)\right]}_{\text{No unknown } r \text{ thanks to Assumption 2}}$

No more r parametrization required. We can avoid the nested optimization by parametrizing Q!

The one-shot minimization by expected risk minimization

Definition 2 (Expected risk minimization problem)

The expected risk minimization problem is defined as the problem of finding Q that minimizes the expected risk $\mathcal{R}_{exp}(Q)$, which is defined as

$$\mathcal{R}_{exp}(Q) := \mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathcal{L}_{NLL}(Q)(s,a) + \mathbb{1}_{a=a_s}\mathcal{L}_{BE}(Q)(s,a)\right]$$

Theorem 3 (Identification via expected risk minimization)

The solution $\widehat{Q}$ to the expected risk minimization problem uniquely identifies Q^* up to $s \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$. Furthermore, we can uniquely identify r up to $s \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$ by

$$r(s,a) = \widehat{Q}(s,a) - \beta \cdot E_{s_{t+1}\sim P(s_t,a_t)}\left[\ln\left(\sum_{a\in\mathcal{A}} \exp\left(\widehat{Q}\left(s_{t+1},a\right)\right)\right) \mid s_t, a_t\right]$$

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Estimation challenge and the TD method

The idea of estimating and minimizing

$$\mathcal{R}_{exp}(Q) := \mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathcal{L}_{NLL}(Q)(s,a) + \mathbb{1}_{a=a_s}\mathcal{L}_{BE}(Q)(s,a)\right]$$

looks straightforward, but it is challenging due to $\mathcal{L}_{BE}(Q)(s,a) = (\mathcal{T}Q(s,a) - Q(s,a))^2$ term

- $\mathcal{T}Q(s,a) := r(s,a) + \beta\mathbb{E}_{s'\sim P(s,a)}\left[\ln\left(\sum_{a\in\mathcal{A}}\exp\left(Q\left(s',a'\right)\right)\right) \mid s,a\right]$ is not empirically available unless we know the transition function
- Instead, $\widehat{\mathcal{T}}Q(s,a,s') := r(s,a) + \beta\left[\ln\left(\sum_{a\in\mathcal{A}}\exp\left(Q\left(s',a'\right)\right)\right) \mid s,a\right]$ is available.

Therefore, a natural proxy for $\mathcal{L}_{BE}(Q)(s,a) = (\mathcal{T}Q(s,a) - Q(s,a))^2$ term is

$$\mathbb{E}_{s'\sim P(s,a)}\left[\mathcal{L}_{TD}(Q)(s,a,s')\right] = \mathbb{E}_{s'\sim P(s,a)}[(\widehat{\mathcal{T}}Q\left(s,a,s'\right) - Q(s,a))^2]$$

This is often called the Temporal difference (TD) method. Would this work?

TD method does not minimize the Bellman error

Issue: The TD method does not guarantee that the Bellman equation holds due to bias.[13]

Lemma 4 (TD is biased)

$$\mathbb{E}_{s'\sim P(s,a)}\left[\mathcal{L}_{TD}(Q)(s,a,s')\right] = \mathcal{L}_{BE}(Q)(s,a) + \underbrace{\mathbb{E}_{s'\sim P(s,a)}\left[(\mathcal{T}Q(s,a) - \hat{\mathcal{T}}Q(s,a,s'))^2\right]}_{\textit{Never 0 unless } P \textit{ is deterministic}}$$

Define $\delta_Q(s,a,s') := \widehat{\mathcal{T}}Q(s,a,s') - Q(s,a)$. Intuitively, the bias above happens because

$$\mathbb{E}_{s'\sim P(s,a)}\left[\delta_Q(s,a,s')^2 \mid s,a\right] \neq \mathbb{E}_{s'\sim P(s,a)}\left[\delta_Q(s,a,s') \mid s,a\right]^2$$

[13] For the same reason, the semi-gradient TD method proposed by Adusumilli and Eckardt (2022) does not guarantee the Bellman equation to hold.

The bi-conjugate trick

We employ an approach often referred to as the "bi-conjugate Trick" which replaces the problematic square function by a linear function called the bi-conjugate:

$$\begin{aligned}\mathcal{L}_{BE}(s,a)(Q) &:= \mathbb{E}_{s'\sim P(s,a)}\left[\delta_Q\left(s,a,s'\right) \mid s,a\right]^2 \\ &= \max_{h\in\mathbb{R}} 2\cdot \mathbb{E}_{s'\sim P(s,a)}\left[\delta_Q\left(s,a,s'\right) \mid s,a\right]\cdot h - h^2\end{aligned}$$

By further re-parametrizing h using $\zeta = h - r(s,a) + Q(s,a)$, after some algebra, we arrive at Lemma 5. (Next page)

Debiasing by bi-conjugate trick

Lemma 5

(a) We can express the squared Bellman error as

$$\begin{aligned}\mathcal{L}_{BE}(Q)(s,a) &:= (\mathcal{T}Q(s,a) - Q(s,a))^2 \\ &= \mathbb{E}_{s'\sim P(s,a)}\left[\mathcal{L}_{TD}(s,a,s')(Q)\right] - \beta^2 D(Q)(s,a)\end{aligned}$$

where, with $V_Q(s) := \ln\left[\sum_{a\in\mathcal{A}} \exp\left(Q(s,a)\right)\right]$,

$$D(Q)(s,a) := \underbrace{\min_{\zeta\in\mathbb{R}} \mathbb{E}_{s'\sim P(s,a)}\left[\left(V_Q\left(s'\right) - \zeta\right)^2 \mid s,a\right]}_{\text{Variance of } V_Q(s') \text{ due to } P}$$

(b) Define the minimizer (over all states and actions) of $D(Q)(s,a)$ *as*

$$\zeta^* : (s,a) \mapsto \underbrace{\arg\min_{\zeta\in\mathbb{R}} \mathbb{E}_{s'\sim P(s,a)}\left[\left(V^*(s') - \zeta\right)^2 \mid s,a\right]}_{\mathbb{E}_{s'\sim P(s,a)}[V^*(s')]}$$

then $r(s,a) = Q^*(s,a) - \beta\zeta^*(s,a)$.

Modified one-shot minimization framework

Theorem 6

Q^* is uniquely identified by

$$\begin{aligned}
&\min_{Q\in\mathcal{Q}} \mathcal{R}_{exp}(Q) \\
&= \min_{Q\in\mathcal{Q}} \mathbb{E}_{(s,a)\sim\pi^*,\nu_0}\left[\mathcal{L}_{NLL}(Q)(s,a) + \mathbb{1}_{a=a_s}\left\{\mathbb{E}_{s'\sim P(s,a)}\left[\mathcal{L}_{TD}(Q)(s,a,s')\right] - \beta^2 D(Q)(s,a)\right\}\right. \\
&= \min_{Q\in\mathcal{Q}} \max_{\zeta\in\mathbb{R}^{S\times A}} \mathbb{E}_{(s,a)\sim\pi^*,\nu_0,s'\sim P(s,a)}\Big[\underbrace{-\log\left(\hat{p}_Q(a\mid s)\right)}_{1)} + \mathbb{1}_{a=a_s}\Big\{\underbrace{\left(\hat{\mathcal{T}}Q(s,a,s') - Q(s,a)\right)^2}_{2)} \\
&\quad - \beta^2 \underbrace{\left(\left(V_Q(s') - \zeta(s,a)\right)^2\right)}_{3)}\Big\}\Big]
\end{aligned} \tag{3}$$

Furthermore, $r(s,a) = Q^*(s,a) - \beta\zeta^*(s,a)$ where ζ^* is defined in Lemma 5.

That is, r estimation is free after Q^* estimation.

Empirical Risk Minimization (ERM) framework

Definition 3 (Empirical risk minimization)

Given $N := |\mathcal{D}|$ where $\mathcal{D}$ is a finite dataset. An empirical risk minimization problem is defined as the problem of finding Q that minimizes the empirical risk $\mathcal{R}_{emp}(Q;\mathcal{D})$, which is defined as

$$\mathcal{R}_{emp}(Q;\mathcal{D}) := \max_{\zeta \in \mathbb{R}^{S\times A}} \frac{1}{N} \sum_{(s,a,s')\in\mathcal{D}}$$

$$\left[-\log\left(\hat{p}_Q(a \mid s)\right) + \mathbb{1}_{a=a_s}\left\{\left(\hat{\mathcal{T}}Q\left(s,a,s'\right) - Q(s,a)\right)^2 - \beta^2\left(V_Q\left(s'\right) - \zeta(s,a)\right)^2\right\}\right]$$

$$= \frac{1}{N}\Big[\sum_{(s,a,s')\in\mathcal{D}} \left(-\log\left(\hat{p}_Q(a \mid s)\right)\right) + \mathbb{1}_{a=a_s}$$

$$\Big(\sum_{(s,a,s')\in\mathcal{D}} \left(\hat{\mathcal{T}}Q\left(s,a,s'\right) - Q(s,a)\right)^2 - \beta^2 \min_{\zeta\in\mathbb{R}^{S\times A}} \sum_{(s,a,s')\in\mathcal{D}} \left(V_Q\left(s'\right) - \zeta(s,a)\right)^2\Big)\Big] \tag{4}$$

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

Bias correction for estimation and the empirical risk minimization framework

GLADIUS algorithm and its global optimality

Simulation experiments

Parametrization and realizability

Given the compact state space $\mathcal{S}$ with $\dim(\mathcal{S}) < \infty$ and the finite action space $\mathcal{A}$ with $|\mathcal{A}| < \infty$, consider a set of parametrized functions

$$\mathcal{Q} = \left\{ Q_{\boldsymbol{\theta}} : \mathbb{R}^{\dim(\mathcal{S})+\dim(\mathcal{A})} \to \mathbb{R} \mid \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^{d_\theta}, Q_{\boldsymbol{\theta}} \in \mathcal{F} \right\}$$

where $\mathcal{F}$ denotes a class of functions such as linear, polynomial or deep neural network function class that is parametrized by $\boldsymbol{\theta}$. We make the following assumption, often called the *realizability* assumption.

Assumption 3 (Realizability)

$\mathcal{Q}$ contains an optimal function Q^*, meaning there exists $\boldsymbol{\theta}^* \in \Theta$ such that $Q_{\boldsymbol{\theta}^*} = Q^*$.

Solving minimax optimization via gradient ascent-descent

We would like to solve the minimax optimization problem

$$\min_{Q\in\mathcal{Q}} \frac{1}{N}\Big[\sum_{(s,a,s')\in\mathcal{D}} \big(-\log\left(\hat{p}_Q(a \mid s)\right)\big) + \mathbb{1}_{a=a_s}$$
$$\Big(\sum_{(s,a,s')\in\mathcal{D}} \left(\hat{\mathcal{T}}Q\left(s,a,s'\right) - Q(s,a)\right)^2 - \beta^2 \min_{\zeta\in\mathbb{R}^{S\times A}} \sum_{(s,a,s')\in\mathcal{D}} \left(V_Q\left(s'\right) - \zeta(s,a)\right)^2\Big)\Big]$$

A canonical way of solving minimax optimization is to apply gradient ascent-descent[14]

- Initialize with arbitrary ζ_{θ_1} and Q_{θ_2}
- Alternative between maximizing for ζ_{θ_1} and minimizing for Q_{θ_2}

[14] Yang et al., "Global convergence and variance-reduced optimization for a class of nonconvex-nonconcave minimax problems"

Algorithm 1 **G**radient-based **L**earning with **A**scent-**D**escent for **I**nverse **U**tility learning from **S**amples (GLADIUS)

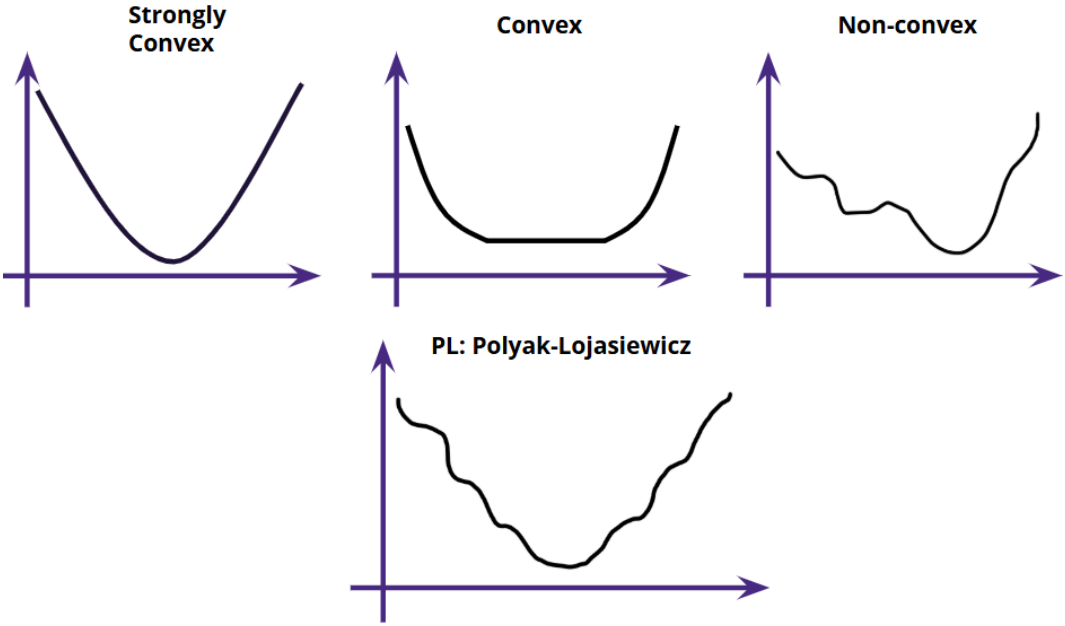
Require: Offline dataset $\mathcal{D} = \{(s, a, s')\}$, time horizon T
Ensure: $\widehat{r}, \widehat{Q}$

Initialize $Q_{\boldsymbol{\theta_2}}, \zeta_{\boldsymbol{\theta_1}}$, iteration $\leftarrow 1$
while $t \leq T$ **do**
  Draw batches B_1, B_2 from $\mathcal{D}$
  [Ascent Step: Update $\zeta_{\boldsymbol{\theta_1}}$, fixing $Q_{\boldsymbol{\theta_2}}$ and $V_{\boldsymbol{\theta_2}}$]
  $D_{\boldsymbol{\theta_1}} \leftarrow \sum_{(s,a,s') \in B_2} \left(V_{\boldsymbol{\theta_2}}(s') - \zeta_{\boldsymbol{\theta_1}}(s, a)\right)^2$
  where $V_{\boldsymbol{\theta}}(s') := \log \sum_{\tilde{a} \in \mathcal{A}} \exp(Q_{\boldsymbol{\theta}}(s', \tilde{a}))$
  $\boldsymbol{\theta_1} \leftarrow \boldsymbol{\theta_1} - \tau_1 \nabla_{\boldsymbol{\theta_1}} D_{\boldsymbol{\theta_1}}$
  [Descent Step: Update $Q_{\boldsymbol{\theta_2}}$ and $V_{\boldsymbol{\theta_2}}$, fixing $\zeta_{\boldsymbol{\theta_1}}$]
  $\overline{\mathcal{L}_{NLL}} \leftarrow \sum_{(s,a,s') \in B_2} -\log\left(\hat{p}_{\boldsymbol{\theta_2}}(a \mid s)\right)$
  $\overline{\mathcal{L}_{\text{BE}}} \leftarrow \sum_{(s,a,s') \in B_1} \mathbb{1}_{a=a_s^*} \left[\mathcal{L}_{TD}(Q)\,(s, a, s') - \beta^2 (V_{\boldsymbol{\theta_2}}(s') - \zeta_{\boldsymbol{\theta_1}}(s, a))^2\right]$
  where $\mathcal{L}_{TD}(Q)\,(s, a, s') := \left(\hat{\mathcal{T}} Q\,(s, a, s') - Q(s, a)\right)^2$
  $\mathcal{L}_{\boldsymbol{\theta_2}} \leftarrow \overline{\mathcal{L}_{\text{NLL}}} + \overline{\mathcal{L}_{\text{BE}}}$
  $\boldsymbol{\theta_2} \leftarrow \boldsymbol{\theta_2} - \tau_1 \nabla_{\boldsymbol{\theta_2}} \mathcal{L}_{\boldsymbol{\theta_2}}$
  iteration $\leftarrow$ iteration $+ 1$
end while
$\widehat{\zeta} \leftarrow \zeta_{\boldsymbol{\theta_1}}$
$\widehat{Q} \leftarrow Q_{\boldsymbol{\theta_2}}$
$\widehat{r}(s, a) \leftarrow \widehat{Q}(s, a) - \beta \cdot \widehat{\zeta}(s, a)$

When is GLADIUS globally optimal?

Definition 4 (Polyak-Łojasiewicz (PL) condition with respect to ℓ_2 norm)

Given $\Theta \in \mathbb{R}^{d_\theta}$, a function $g : \Theta \mapsto \mathbb{R}$ is said to satisfy the Polyak-Łojasiewicz (PL) condition with respect to ℓ_2 norm if g has a nonempty solution set and a finite minimal value $g(\boldsymbol{\theta}^*)$ for $\boldsymbol{\theta}^* \in \Theta \subseteq \mathbb{R}^d$, and there exists some $c > 0$ such that $\frac{1}{2}\|\nabla g(\boldsymbol{\theta})\|_2^2 \geq c(g(\boldsymbol{\theta}) - g(\boldsymbol{\theta}^*)), \forall \boldsymbol{\theta} \in \Theta$.


Strongly Convex
Convex
Non-convex
PL: Polyak-Lojasiewicz

When is GLADIUS globally optimal?

Lemma 7 (Yang et al., 2020)

A minimax optimization of function $f(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ (minimizing for $\boldsymbol{\theta}_1$, maximizing for $\boldsymbol{\theta}_2$) is guaranteed to converge to a global minimum through gradient ascent-descent if the function satisfies the PL condition in terms of both $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$.

Corollary 8

GLADIUS achieves global optimality if

$$\frac{1}{N}\Big[\sum_{(s,a,s')\in\mathcal{D}} \big(-\log\big(\hat{p}_{Q_{\theta_2}}(a \mid s)\big)\big) + \mathbb{1}_{a=a_s}$$

$$\Big(\sum_{(s,a,s')\in\mathcal{D}} \big(\hat{\mathcal{T}} Q_{\theta_2}(s,a,s') - Q_{\theta_2}(s,a)\big)^2 - \beta^2 \sum_{(s,a,s')\in\mathcal{D}} \big(V_{Q_{\theta_2}}(s') - \zeta_{\theta_1}(s,a)\big)^2\Big)\Big]$$

satisfies PL in terms of both $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$.

Is GLADIUS globally optimal?

Lemma 9

For linear function and a Neural network parametrization of Q as $Q_{\boldsymbol{\theta}}$, $\mathcal{L}_{NLL}(Q_{\boldsymbol{\theta}})(s,a)$ and $\mathcal{L}_{BE}(Q_{\boldsymbol{\theta}})(s,a)$ satisfies PL condition with respect to $\boldsymbol{\theta}$.

Lemma 10

For linear function and a Neural network parametrization of Q and ζ, the expected risk and the empirical risk both satisfy the PL condition with respect to $\boldsymbol{\theta_1}$ and $\boldsymbol{\theta}_2$.

Theorem 11 (Global optimality of GLADIUS)

For linear function and a Neural network parametrization of Q and ζ, the above equation satisfies PL for both $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. That is, GLADIUS achieves global optimality.

Statistical analysis

Lemma 12 (Charles and Papailiopoulos (2018), Lei (2023))

Define N-sampled dataset as $\mathcal{D}_N$. Also, define the minimum of empirical risk $\mathcal{R}_{emp}(\boldsymbol{\theta})$ as $\boldsymbol{\theta}^*_{\mathcal{D}_N}$. If $\mathcal{R}_{emp}(\boldsymbol{\theta})$ satisfies PL in terms of $\boldsymbol{\theta}$,

$$\mathbb{E}_{\mathcal{D}_{\mathcal{N}}}\left[\left(\boldsymbol{\theta}^*_{\mathcal{D}_N} - \boldsymbol{\theta}^*\right)^2\right] \leq \frac{2\sigma}{N}$$

where σ is specified in the paper.

Theoretical Results for GLADIUS – A Summary

Theorem 13 (Global convergence)

Under Assumption 3, for linear and neural network parametrization of Q and ζ,

$$\mathbb{E}_{\mathcal{D}_N}\left[\left(\theta^* - \hat{\theta}_T\right)^2\right] \leq \frac{2\nu}{\gamma + T} + \frac{2\sigma}{N}$$

where T is the number of gradient steps, N is the sample size, and the values of constants γ, ν, and σ are specified in the paper.

This is $O(1/\sqrt{T})$ and $O(1/\sqrt{N})$ rate convergence (par existing parametric methods)

Outline

Motivation: Large-scale problems

Recap: Dynamic Discrete Choice (DDC) models

Recap: what we know about solving the DDC models

The one-shot gradient-based optimization approach

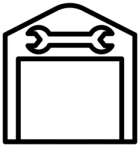
Bias correction for estimation and the empirical risk minimization framework

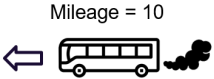
GLADIUS algorithm and its global optimality

Simulation experiments

Simulation

A standard econometrics benchmark: Rust (1987) bus engine replacement simulation





Simulation

A standard econometrics benchmark: Rust (1987) bus engine replacement simulation

Engine replacement

v.s.

Engine maintenance

Replacement cost = $\text{mileage} * c_1$

Maintenance cost = c_2

Simulation

A standard econometrics benchmark: Rust (1987) bus engine replacement simulation

Engine replacement

Mileage = 0 (reset)

Engine maintenance

Mileage = 10 (continued)

(when bus returns, mileage randomly increases)

Performance test

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC

Performance test

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC

"Oracle" baselines use

1. Linear reward model: Replacement cost = $\text{mileage} * c_1$
   Maintenance cost = c_2

2. Exact transition probabilities

Performance test

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC

Developed by Amazon;
Q parametrization + TD

Performance test

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC

Popular IRL algorithms

Performance test

Mean Absolute Percentage Error (MAPE) $\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{r}_i - r_i}{r_i}\right| \times 100$

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC
	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)

-

Performance test

Mean Absolute Percentage Error (MAPE) $\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{r}_i - r_i}{r_i}\right| \times 100$

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC
	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)
50						
250						
500						
1000						
2500						
5000						

Based on 20 repetitions. Oracle baselines (Rust, MLIRL) were based on bootstrap repetition of 100.

Performance test

Mean Absolute Percentage Error (MAPE) $\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{r}_i - r_i}{r_i}\right| \times 100$

No. of Trajectories (H=100)	Oracle Baselines		Nonparametric, No Knowledge of Transition Probabilities			
	Rust	ML-IRL	**Ours**	SAmQ	IQ-learn	BC
	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)
50	3.62 (1.70)	3.62 (1.74)	**3.44** (1.28)	4.92 (1.20)	114.13 (26.60)	80.55 (12.82)
250	1.37 (0.77)	1.10 (0.78)	**0.84** (0.51)	3.65 (1.00)	112.86 (27.31)	72.04 (13.21)
500	0.90 (0.56)	0.84 (0.59)	**0.55** (0.20)	3.13 (0.86)	113.27 (25.54)	71.92 (12.44)
1000	0.71 (0.49)	0.64 (0.48)	**0.52** (0.22)	1.55 (0.46)	112.98 (24.12)	72.17 (12.11)
2500	0.68 (0.22)	0.62 (0.35)	**0.13** (0.06)	N/A	111.77 (23.99)	62.61 (10.75)
5000	0.40 (0.06)	0.43 (0.26)	**0.12** (0.06)	N/A	119.18 (22.55)	46.45 (8.22)

Based on 20 repetitions. Oracle baselines (Rust, MLIRL) were based on bootstrap repetition of 100.

Why does this happen?

Reason 1. GLADIUS does well only in the frequently occurring regions

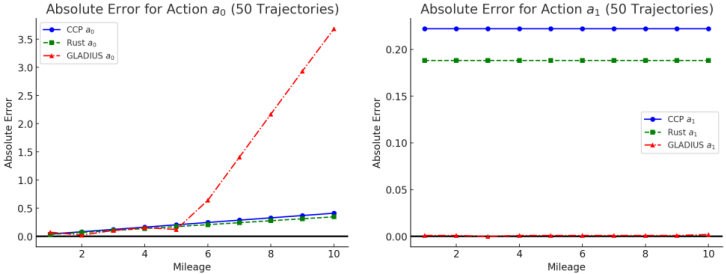


Figure 2: Reward estimation error comparison for 50 trajectories. Closer to 0 (black line) is better.

Mileage	Frequency		Ground Truth r		ML-IRL		Rust		GLADIUS	
	a_0	a_1	a_0	a_1	a_0	a_1	a_0	a_1	a_0	a_1
1	412	37	-1.000	-5.000	-0.959	-4.777	-0.965	-4.812	-1.074	-4.999
2	65	18	-2.000	-5.000	-1.918	-4.777	-1.931	-4.812	-1.978	-5.001
3	43	80	-3.000	-5.000	-2.877	-4.777	-2.896	-4.812	-3.105	-5.000
4	24	101	-4.000	-5.000	-3.836	-4.777	-3.861	-4.812	-3.844	-5.001
5	8	134	-5.000	-5.000	-4.795	-4.777	-4.827	-4.812	-4.878	-5.001
6	4	37	-6.000	-5.000	-5.753	-4.777	-5.792	-4.812	-6.642	-5.001
7	1	26	-7.000	-5.000	-6.712	-4.777	-6.757	-4.812	-8.406	-5.001
8	0	7	-8.000	-5.000	-7.671	-4.777	-7.722	-4.812	-10.170	-5.001
9	0	2	-9.000	-5.000	-8.630	-4.777	-8.688	-4.812	-11.934	-5.001
10	0	1	-10.000	-5.000	-9.589	-4.777	-9.653	-4.812	-13.684	-5.002

Table 5: Reward estimation for 50 trajectories. Color indicates appearance frequencies.

Reason 2. Estimation difficulty under data imbalance

This is a well-known issue in RL [15]: "*The use of a projection operator onto a space of function approximators with respect to a distribution induced by the behavior policy can result in poor performance if that distribution does not sufficiently cover the state space.*".

- i.e., extrapolation to data-poor regions = ☹
- All trajectories start from mileage 1. Also, the replacement action (action 0) resets the mileage to 1. Therefore, most states observed in the expert data are within mileage 1-5.

[15] Tsitsiklis and Van Roy, "Analysis of temporal-difference learning with function approximation", NeurIPS 1996

Scalability

A standard econometrics benchmark: Rust (1987) bus engine replacement simulation

What if we don't know mileage only matters?

Bus mileage at $t = 1$

Bus age at $t = 1$

Downtime cost

Customer complaints

Engine temperature

⋮

More dummy variables

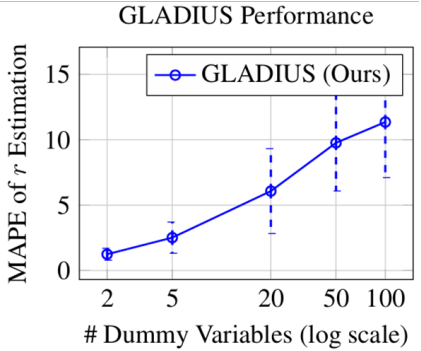
⋮

Scalability test result

Mean Absolute Percentage Error (MAPE) $\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{r}_i - r_i}{r_i}\right| \times 100$

Dummy Variables	**Ours**	SAmQ	IQ-learn	BC
	MAPE (SE)	MAPE (SE)	MAPE (SE)	MAPE (SE)
2	**1.24** (0.45)	1.79 (0.37)	112.0 (14.8)	150.9 (29.1)
5	**2.51** (1.19)	2.77 (0.58)	192.2 (19.2)	171.1 (37.3)
20	**6.07** (3.25)	N/A	180.1 (15.6)	180.0 (33.7)
50	**9.76** (3.68)	N/A	282.2 (25.2)	205.1 (35.3)
100	**11.35** (4.24)	N/A	321.1 (23.1)	288.8 (42.9)

Based on 10 repetitions. For SAmQ, N/A means that the algorithm did not scale.



Summary

- We want a scalable DDC method. Combine with non-parametric (e.g., ML) methods?
- Fitted fixed-point iteration methods are provably unstable.
- No globally optimal convergent gradient-based method exists.
- In this study, we propose one!
  - Empirical Risk Minimization (ERM) framework
  - Global optimally convergent, e.g., even with Neural Network parametrization
  - Nice empirical performance for standard benchmark
  - Scales well up to 10^{100} states ($\gg$ # of atoms in the universe), losing only 10% precision

Further discussions

- Q: How is it possible to identify the rewards if the data size is far smaller than 10^{100} states?

  A: Think of it like fitting a smooth sheet to a few measured points on a landscape: once you know how the sheet bends, you can read off the height anywhere without having measured every spot. The sheet is controlled by only a few-thousand adjustable "knobs" (neural network weights). Sample size we need depends on the knob structure complexity.

- Q: Given $\hat{r}$, don't we anyway need explicit estimation of P for counterfactual analysis?

  - A: No, given dataset $\{s_0, a_0, s_1, a_1 \ldots\}$, we now have $\{s_0, a_0, \hat{r}(s_0, a_0), s_1, a_1, \hat{r}(s_1, a_1) \ldots\}$. Given a counterfactual scenario (e.g., a reward intervention), we can apply offline RL algorithms directly to recover the corresponding counterfactual optimal policy.

Future extensions

- The proposed method can be extended to address the unobserved heterogeneity.
  - Adapting to unobserved heterogeneity is often called "in-context learning (ICL)" in ML
  - We can implement ICL given a gradient-based estimation method [16] [17]
  - In this paper, we propose one, so we can implement ICL! (A working paper coming soon!)
- The PL condition proof can be extended for offline reinforcement learning (offline RL)
  - No gradient-based offline RL method has been proven to be globally convergent
  - The results in this paper can be easily extended (A working paper coming soon!)

[16] Ahn et al. "Transformers learn to implement preconditioned gradient descent for in-context learning." NeurIPS23

[17] Wang et al. "Transformers learn temporal difference methods for in-context reinforcement learning.", arXiv:2405.13861.

Thank you for listening!

Special thanks to John Rust for the kind invitation to give this talk and for his detailed and insightful comments on the paper. Many thanks as well to the DSE organizers for their excellent coordination.