FORCES
FOUNDATIONS OF RESILIENT CYBER-PHYSICAL SYSTEMS

Berkeley
UNIVERSITY OF CALIFORNIA

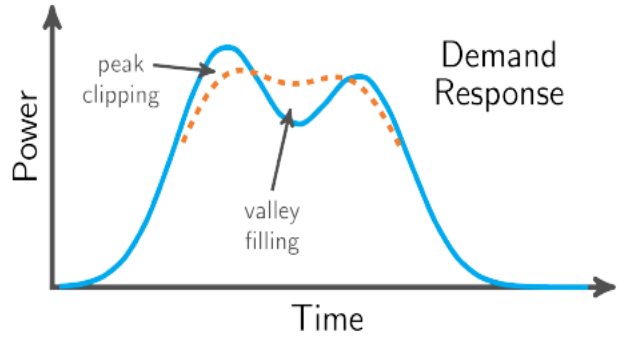
Learning Agent Preferences via Inverse Risk-Sensitive Reinforcement Learning

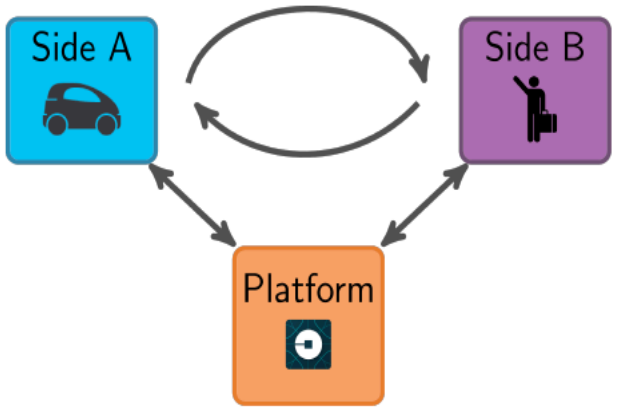
Eric Mazumdar
joint work with Tanner Fiez, Lillian Ratliff, and S. Shankar Sastry
24/08/2017

Learning Models of Human Behavior

With new services and emerging markets, humans are transitioning from *passive* to *active* participants

- People's decisions are increasingly sequential and depend on a number of exogenous and endogenous factors

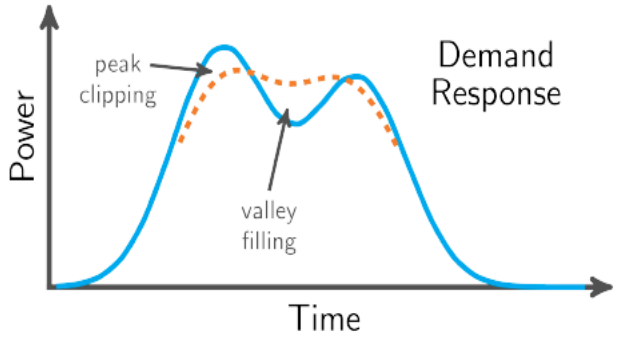


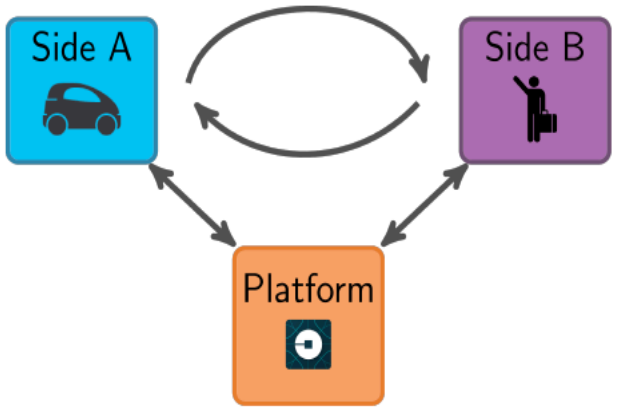


Learning Models of Human Behavior

With new services and emerging markets, humans are transitioning from *passive* to *active* participants

- People's decisions are increasingly sequential, and depend on a number of exogenous and endogenous factors





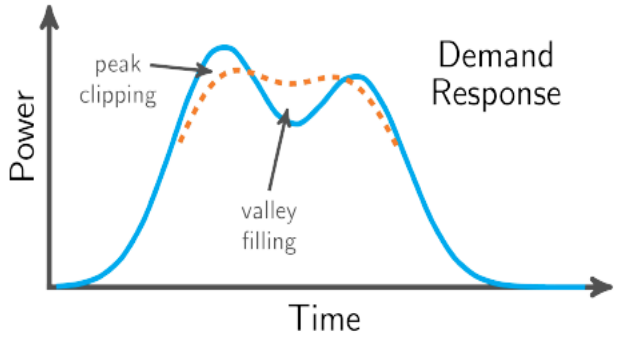
Goal:

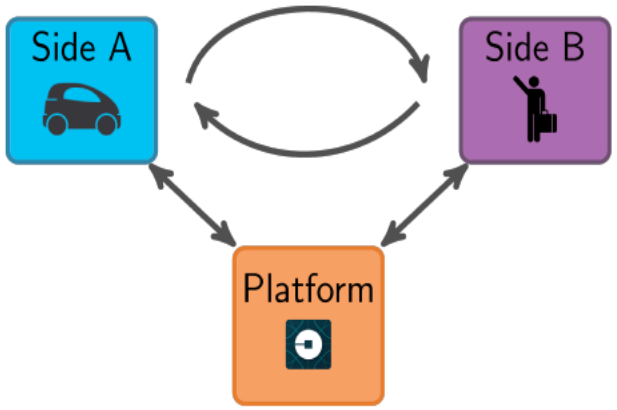
Leverage increasingly fine-grained decision-making data to *learn plausible models* of human behavior and preferences from data

Learning Models of Human Behavior

Goal:

Leverage increasingly fine-grained decision-making data to learn plausible models of human behavior and preferences from data





What do we want out of these models?

- *Accuracy:* for use in control and incentive schemes.
- *Interpretability*: policy makers, system designers, etc. need to be able to understand what the models are capturing.

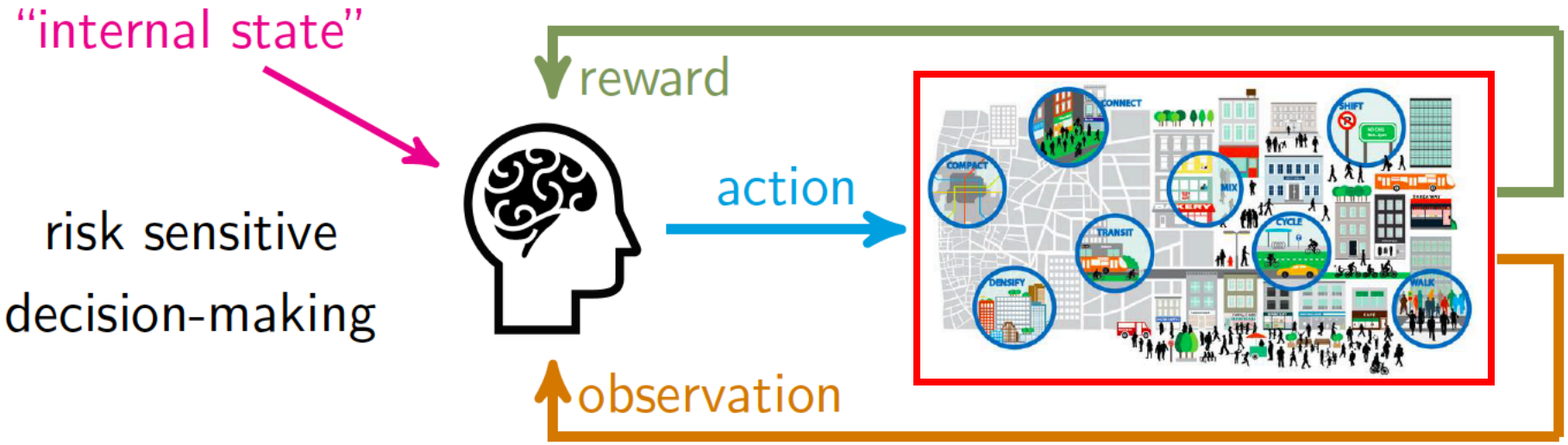
Outline

1. Overview of Risk-Sensitive Decision Making

2. Risk-Sensitive Q-Learning

3. Inverse Risk-Sensitive Reinforcement Learning
   - Problem Formulation
   - Theoretical Results
   - Example: Risk-Sensitive agents in GridWorld
   - Example: Sensitivity to Surge-Pricing in Ride-Sharing

Risk-Sensitive Decision Making



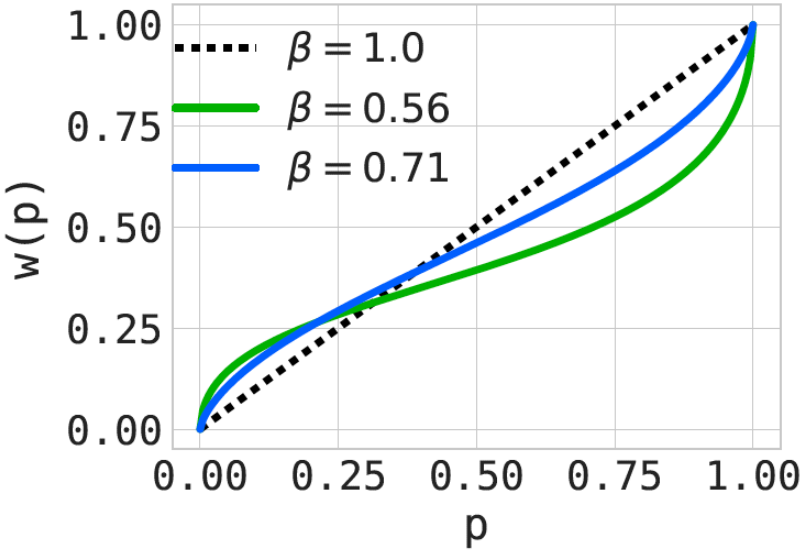
People are *not completely rational agents*

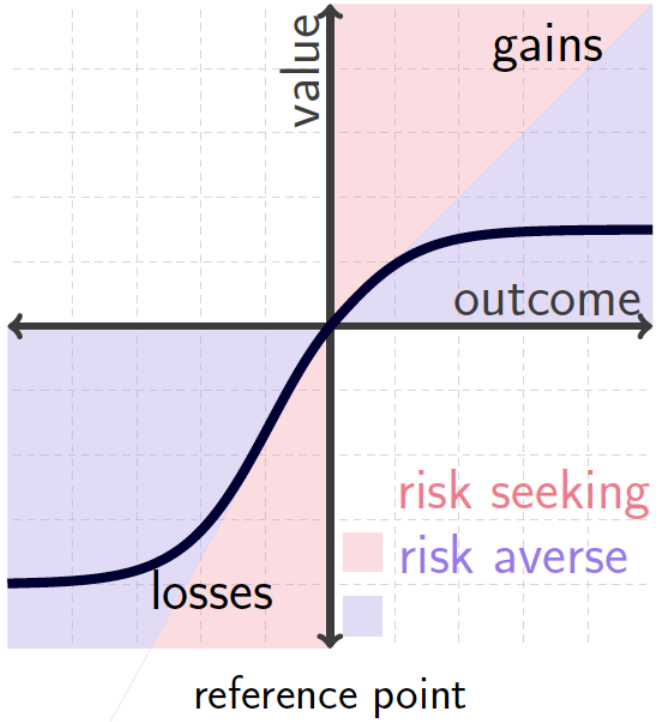
- Treat losses and gains differently
- Make decisions based on:
  - Warped event probabilities
  - Comparisons to *reference points*

[2],[3]

Prospect Theory

1. People tend to warp event probabilities:
   - Overweight low probability events
   - Underweight high probability events
2. People compare outcomes to *reference points*
   - *Outcomes with higher values are "gains" and losses otherwise*
   - *Losses tend to loom larger than gains*
3. *Risk-Attitudes* are impacted by the reference points
   - People are *risk-averse* on gains and *risk-seeking on losses (concave vs. convex) shape*



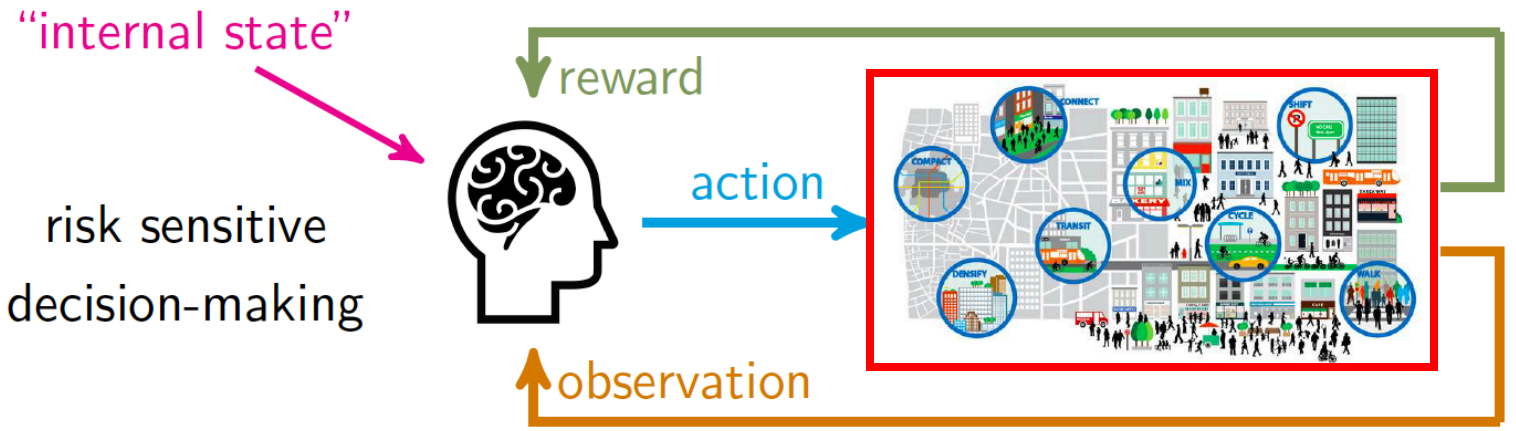


[2]-[4]

Risk-Sensitive Q-Learning

- MDP Framework:
  - We consider a class of finite MDPs {X,A,P,R}, where
    - X -State space
    - A - Action Space
    - P(x'|x,a) - transition Kernel
    - R: $X \times A \to \mathbb{R}$ - reward function

- Agents are Risk-Sensitive and process their observations through their value functions



Value Functions

Prospect Theory:

Incorporates reference point, r_0 , and different risk sensitivities for gains and losses

- Kahneman & Tversky:

$$u(y) = \begin{cases} c_+(y - r_0)^{\rho_+}, & \text{if } y \geq r_0 \\ c_-(y - r_0)^{\rho_-}, & \text{otherwise} \end{cases}$$

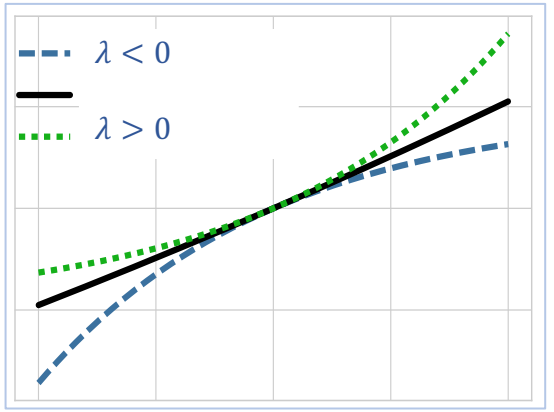
- Log Prospect Function:
  - Lipschitz approximation to classical prospect theory function

$$u(y) = \begin{cases} c_+ \log\left(\rho_+(y - r_0) + 1\right), & \text{if } y \geq r_0 \\ c_- \log\left(\rho_-(r_0 - y) + 1\right), & \text{otherwise} \end{cases}$$

Entropic Map:

- Historically used in economics and finance to model risk-sensitivity, λ determines risk-sensitivity

$$u(y) = \frac{1}{\lambda}(e^{-\lambda y} - 1)$$



[2]-[4]

Valuation Functions

Valuation functions *generalize the expectation* to capture risk-sensitivity:

$\mathcal{V} : \mathbb{R}^{|I|} \times \mathcal{P} \to \mathbb{R}$ is a *Valuation Function* if for each $\mu \in \mathcal{P}$:

1. It is monotonic - $\mathcal{V}_\mu(Y) \leq \mathcal{V}_\mu(Z)$ whenever $Y \leq Z$
2. It is translation invariant - $\mathcal{V}_\mu(Y + c) = \mathcal{V}_\mu(Y) + c$ for any $c \in \mathbb{R}$

In the risk-sensitive MDP formulation, we introduce the valuation map:

$$\mathcal{V}_{x,a}(Y) = \mathcal{V}_{\mu_{x,a}}(Y|x,a) : \mathbb{R}^{|I|} \times X \times A \to \mathbb{R}$$

$$\text{where: } \mu_{x,a}(x') = P(x'|x,a)$$

[4]-[6]

Utility–Based Shortfall Valuation

We use the *utility-based shortfall* valuation map:

- Comparison to an "acceptance level", x_0
- Long been used it mathematical finance, and economics

$$\mathcal{V}_{x,a}(Y) = \sup\{m \in \mathbb{R} | \mathbb{E}_{x,a}[u(Y-m)] \geq x_0\}$$

- For simplicity we often use, $x_0 = u(r_0)$

Risk-Sensitive Q-Learning

We start by defining the *cost-to go* associated with a policy π and initial state x_0:

$$J(\pi, x_0) = \mathcal{V}_{x_0}^{\pi}\left[R(x_0, a_0) + \gamma \mathcal{V}_{x_1}^{\pi}\left[R(x_1, a_1) + \ldots + \gamma \mathcal{V}_{x_T}^{\pi}[R(x_T, a_T) + \ldots]\right]\right]$$

Now we can define the *value function*, V^* of the MDP:

$$\text{If } V^* \text{ satisfies: } V^*(x_0) = \max_{a \in A(x)} \mathcal{V}_{x,a}\left[R(x, a) + \gamma V^*(x')\right]$$

$$\text{then, for all } x \in X\text{: } V^* = \max_{\pi} J(\pi, x)$$

Now, given our value function, we define our *Q-function*: $Q^*(x, a) = \mathcal{V}_{x,a}(R + \gamma V^*)$

$$Q^*(x, a) = \mathcal{V}_{x,a}\left(R + \gamma \max_{a \in A(x')} Q^*(x', a)\right), \quad \forall\ (x, a) \in X \times A$$

[4]-[6]

Risk-Sensitive Q-Learning

Q-Learning

$$Q^*(x,a) = \mathbb{E}_\pi \left[R + \gamma \max_{a' \in A(x')} Q^*(x',a') \right]$$

$$\mathbb{E}[R + \gamma \max_{a \in A(x')} Q^*(x',a) - Q^*(x,a)] = 0$$

Risk Sensitive Q-Learning

$$Q^*(x,a) = \mathcal{V}_{x,a} \left[R + \gamma \max_{a' \in A(x')} Q^*(x',a') \right]$$

$$\mathbb{E}[u(R(x,a) + \gamma \max_{a' \in A} Q^*(x',a') - Q^*(x,a))] = x_0$$

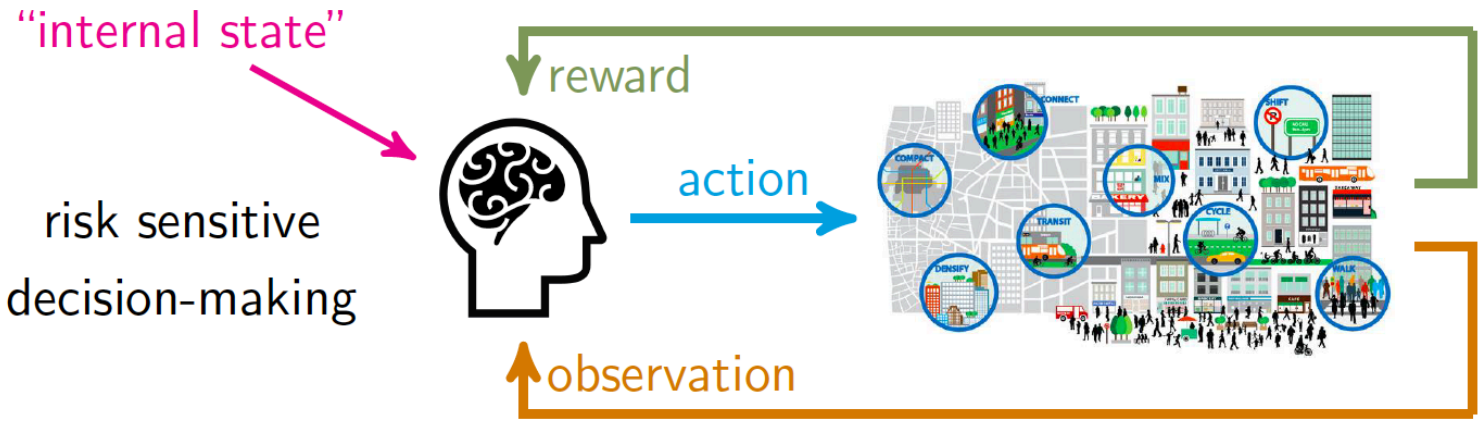
Risk –Sensitive Q-Learning Update: The *value function* is applied to the *temporal differences*

$$Q_{k+1}(x_t,a_t) \leftarrow Q_k(x_t,a_t) + \alpha \mathbb{E}[u(r_t + \gamma \max_a Q_k(x_{t+1},a) - Q_k(x_t,a_t))] - x_0$$

[4]-[6]

Inverse Risk-Sensitive Reinforcement Learning

We assume that the agent in question (the human) is operating according to our forward model:



Goal:

Given observations of an Agent's sequential decisions, we would like to infer the decision-making model of the agent.

We assume that the agent (the human) is operating according to our forward model, meaning:

- Dynamics: Agent operates in $\mathcal{M} = \{X, A, P, R\}$ with discount factor γ.

- Decision Making Model:
  - Agent's value function u, belongs in a parametric class:

$$\{u_{\theta_1}\}_{\theta_1}, \quad u : Y \times \Theta_1 \to \mathbb{R}, \quad u_{\theta_1} : Y \to \mathbb{R}$$

    e.g. class of prospect theoretic value functions where $\theta_1 = \{k_-, k_+, l_-, l_+\}$.

  - The policy, π, is a deterministic function of the Q-function, and belongs in a parametric class $\{\pi_\theta\}_\theta$ e.g. soft-max

$$\pi_\theta(a|x) = G_\theta(x, a) = \frac{exp(\beta Q(x, a, \theta))}{\sum_{a' \in A} exp(\beta Q(x, a', \theta))}$$

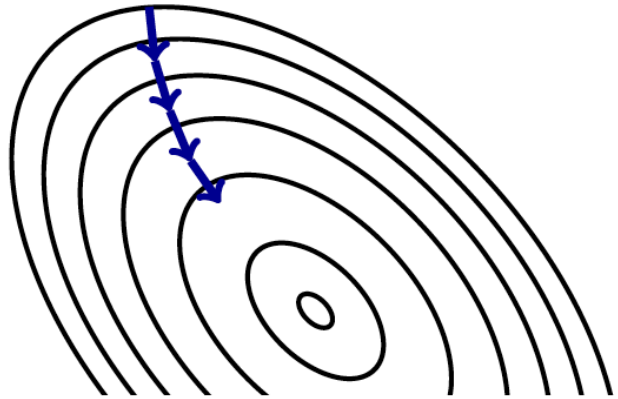
    where $\theta = \{\theta_1, \beta, \gamma\} \in \Theta$

Optimization Formulation

Given the and data from the agent operating in the MDP, $\mathcal{D} = \{(x_k, a_k)\}_{k=1}^{N}$, we would like to tune the parameters θ, to minimize some loss, $\ell(\theta) = \ell(\pi_\theta)$:

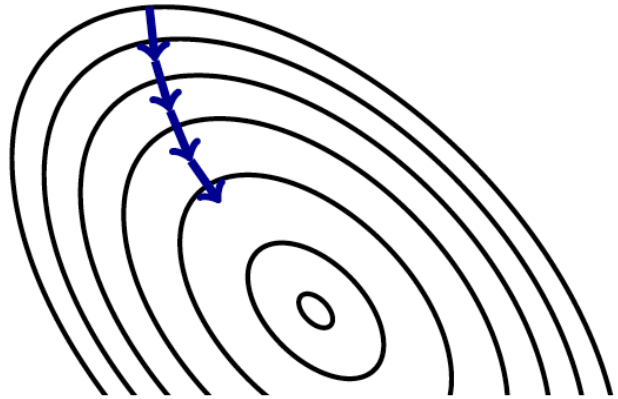
$$\min_{\theta \in \Theta} \{\ell(\pi_\theta) | \pi_\theta = G_\theta(Q^*), u_\theta\}$$

Goal: We would like to solve this using *gradient methods*.
(*Why?* – easy to implement and use local information, large number of approaches, potentially scalable,…)



Optimization Formulation

Goal: We would like to solve this using *gradient methods*. (*Why?* – easy to implement and use local information, large number of approaches, potentially scalable,…)



Challenges:

- Problem is highly non-convex - many parameterizations can yield the same loss

- Computing the derivative of ℓ wrt θ requires computing the derivative of Q^* wrt θ_1
  - in general, $\theta_1 \mapsto Q^*_{\theta_1}$ is non-differentiable

$$D_{\theta_k}\pi_\theta(a|x) = \pi_\theta(a|x) D_{\theta_k} \ln(\pi_\theta(a|x))$$
$$= \pi_\theta(a|x)\beta\left(D_{\theta_k}Q^*(x,a,\theta) - \sum_{a' \in A} \pi_\theta(a'|x) D_{\theta_k} Q^*(x,a',\theta)\right)$$

Optimization Formulation – Gradient Calculation

assumptions on $u : Y \times \Theta \to \mathbb{R}$:

1. continuous, strictly increasing in y & for each θ, $\exists\ y_0,\ u(y_0, \theta) = r_0$
2. it is Lipschitz in y & locally Lipschitz on Θ
3. $\exists\ \varepsilon > 0$ such that $\varepsilon \leq \frac{u(y,\theta) - u(y',\theta)}{y - y'}$

Thm[1,2]: Assume that $u : Y \times \Theta \to \mathbb{R}$ satisfies the above assumptions. Then the following statements hold:

(a) Q^*_θ is locally Lipschitz-continuous as a function of θ

(b) Except on a set of measure zero, the gradient $D_\theta Q^*_\theta$ is given by the solution of the fixed–point equation

$$\begin{aligned}\phi_\theta(x,a) = \alpha \mathbb{E}_{x,a,w} \Big[& D_\theta \tilde{u}(y(\theta),\theta) + D_y \tilde{u}(y(\theta),\theta) \\ & \cdot (\gamma \phi_\theta(x', a^*_{x'}) - \phi_\theta(x,a)) \Big] + \phi_\theta(x,a)\end{aligned}$$

Optimization Formulation – Loss Functions

$$\min_{\theta \in \Theta} \{\ell(\pi_\theta) | \pi_\theta = G_\theta(Q^*), u_\theta\}$$

1. negative log-likelihood: $\ell(\boldsymbol{\theta}) = \sum_{(x,a)\in\mathscr{D}} \log(\boldsymbol{\pi}_{\boldsymbol{\theta}}(x,a))$
2. negative weighted log-likelihood: $\ell(\boldsymbol{\theta}) = \sum_{(x,a)\in\mathscr{D}} w(x,a)\log(\boldsymbol{\pi}_{\boldsymbol{\theta}}(x,a))$
3. relative entropy or KL divergence: $\ell(\boldsymbol{\theta}) = \sum_{x\in\mathscr{D}_x} D_{\mathsf{KL}}(\hat{\boldsymbol{\pi}}(\cdot|x)||\boldsymbol{\pi}_{\boldsymbol{\theta}}(\cdot|x))$

Why one or the other?

- In general, the inverse problem is ill-posed
- The log-likelihood loss function encourages mimicking the observed behaviors (may over fit to states that are over-represented in dataset)
- The KL-divergence prioritizes finding parameters that match the empirical policy across all states uniformly.

Sample Complexity

How can we assess how the performance of our algorithm varies with the amount of data?

- Challenge: We do not have access to the policy of the "true agent" $\pi(\cdot\,|x)$, but to the empirical policy of the agent $\pi_n(\cdot\,|x)$
- Challenge: Non-convexity of the objective function

Using properties of the SLLNs and properties of discrete probability distributions, with probability 1- ρ :

$$||\pi_\theta(\cdot|x) - \pi(\cdot|x)|| \leq |A|\left(\frac{2}{n(x)}\log\frac{2|A|}{\rho}\right)^{1/2} + ||\pi_\theta(\cdot|x) - \pi_n(\cdot|x)||$$

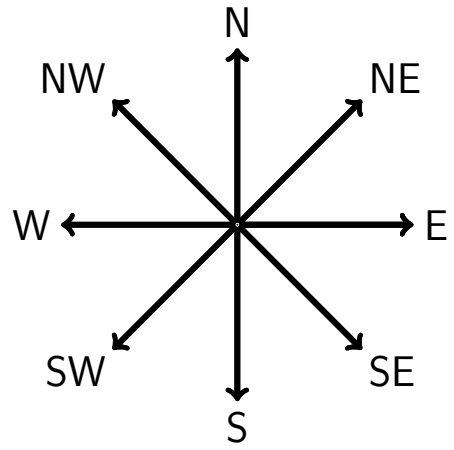
Sample Complexity

$$||\pi_\theta(\cdot|x) - \pi(\cdot|x)|| \leq \underbrace{|A|\left(\frac{2}{n(x)}\log\frac{2|A|}{\rho}\right)^{1/2}}_{\text{Depends only on data}} + \underbrace{||\pi_\theta(\cdot|x) - \pi_n(\cdot|x)||}_{\text{Depends only on training error}}$$

- This gives us a bound, per state, on how well the learned value function can recreate the behavior.
- It also implicitly tells you, what states you are most uncertain about and therefore which states you should collect more data from.
- Thus, at best, the convergence of the policy under the learned value function to the ”true” policy is $O\left(\frac{1}{\sqrt{n}}\right)$, where n is the number of trajectories sampled from the agent.

Example 1: Risk-Sensitive Agents in Grid World

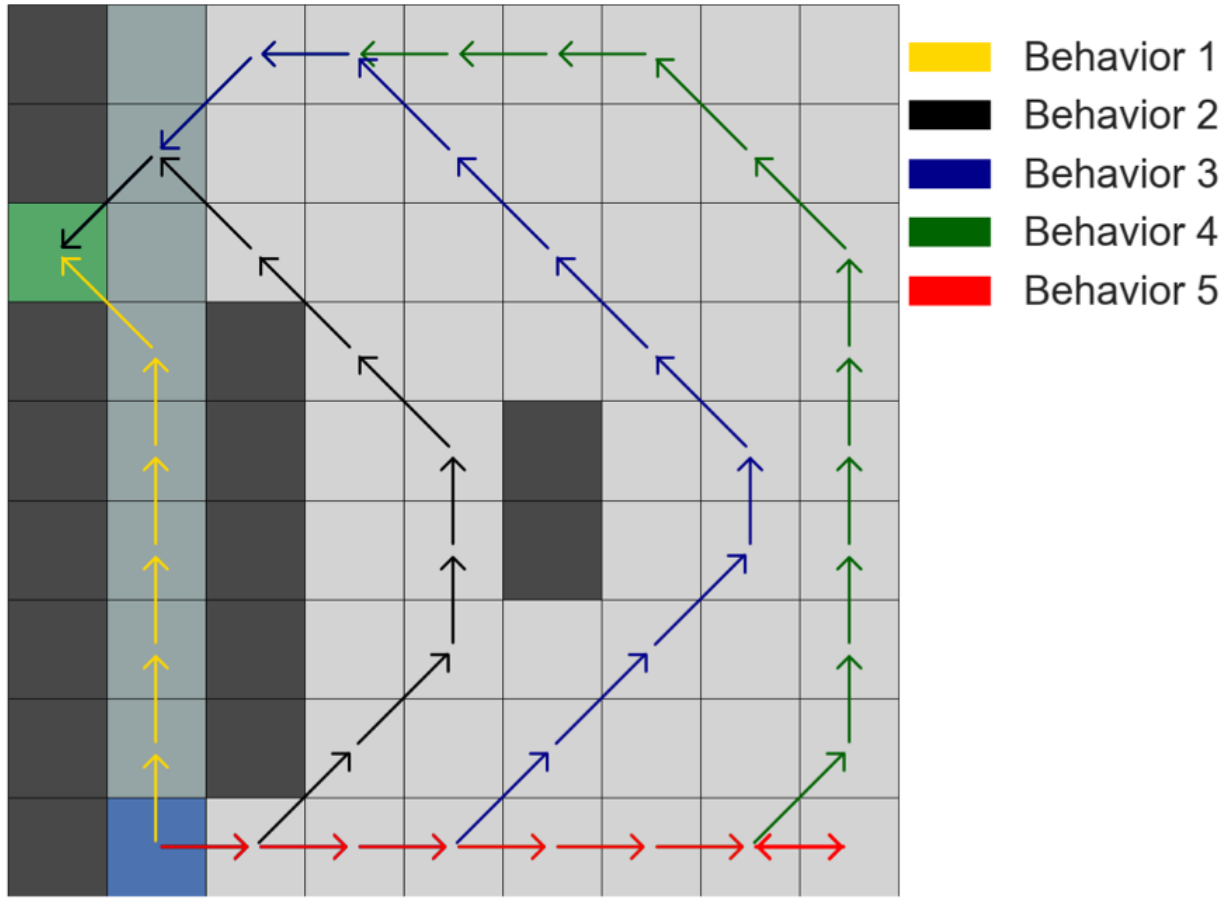
-1	-0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
+1	-0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	-1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	-1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	start	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1



Dynamics:

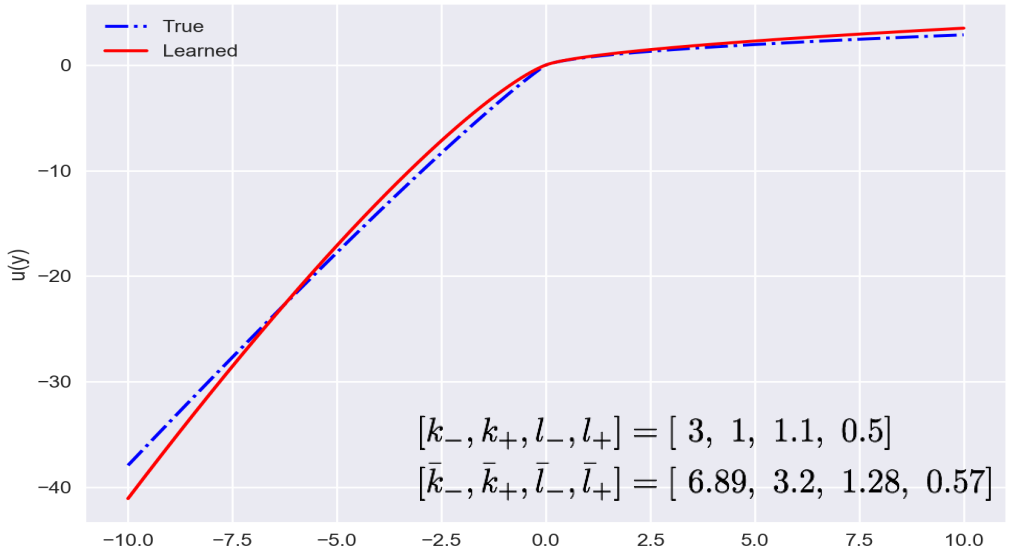
- Black and Green states are absorbing
- The agent moves in their desired direction with probability 0.93 and a random other direction with probability 0.07

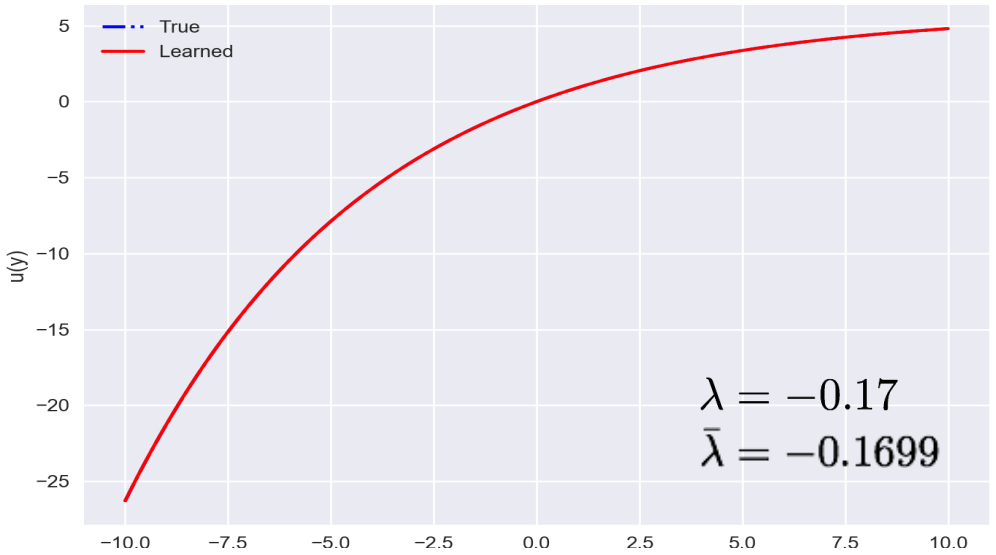
-1	-0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
+1	-0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	-1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	-1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	-0.1	-1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1
-1	start	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1	+0.1

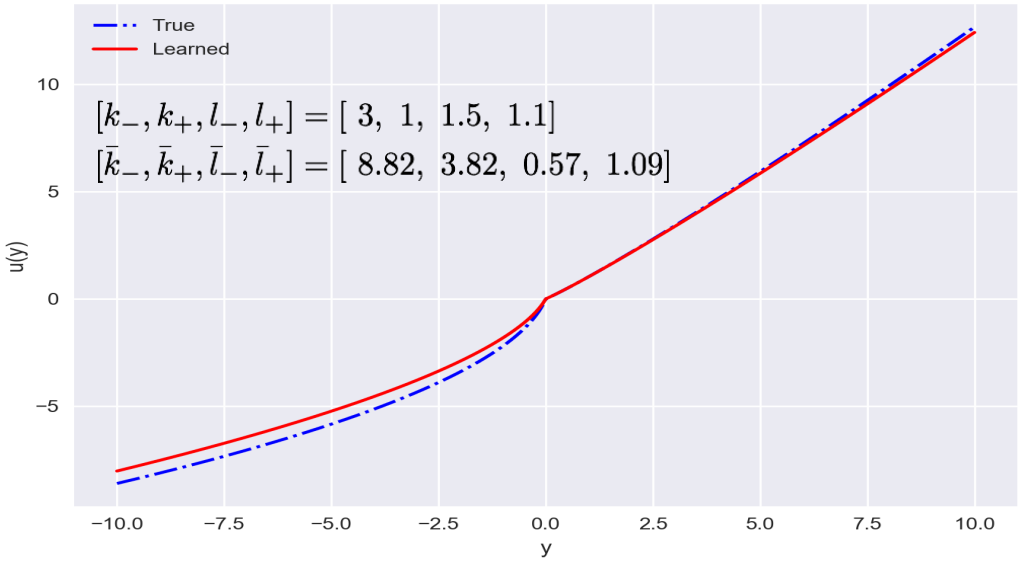


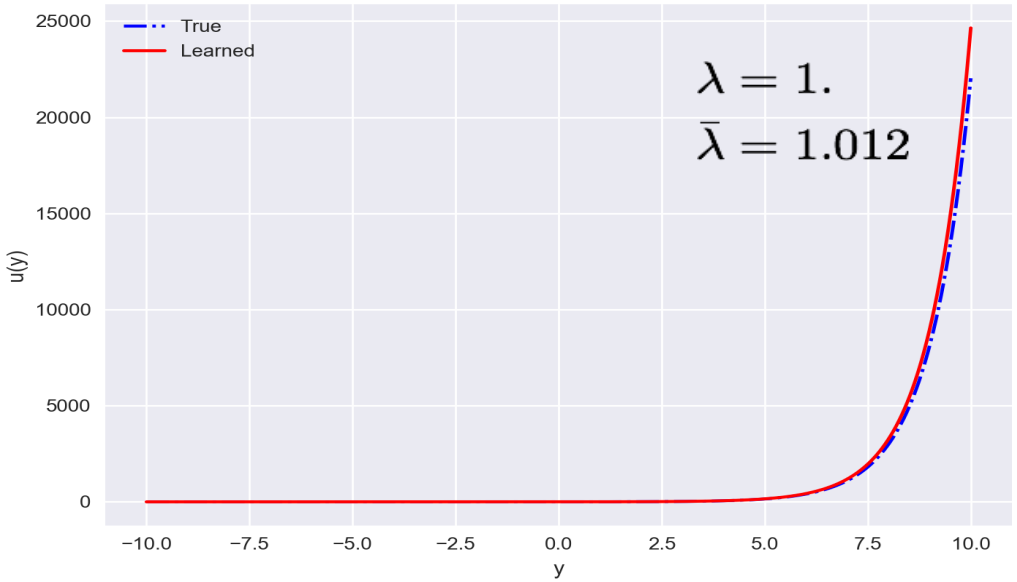
Value Function	*Prospect*		*L-Prospect*		*Logarithmic*		*Entropic*	
Behavior	Mean	Var	Mean	Var	Mean	Var	Mean	Var
1	0.019	6.3e−4	0.013	2.3e−4	0.014	2.7e−4	1.6e−3	5.1e−6
2	0.015	2.0e−4	0.010	9.6e−5	4.0e−3	1.6e-5	2.6e−4	1.4e−7
3	0.020	3.6e−4	0.011	1.3e−4	0.036	2.9e-3	2.2e−3	1.5e−5
4	0.016	2.0e−4	0.012	1.4e−4	0.084	0.014	4.6e−4	1.8e−7
5	0.047	3.0e−3	0.010	3.4e−4	0.14	0.033	6.6e−4	2.2e−7

Even though the problem is ill-posed, the *true* and *learned* value functions are similar when scaled.









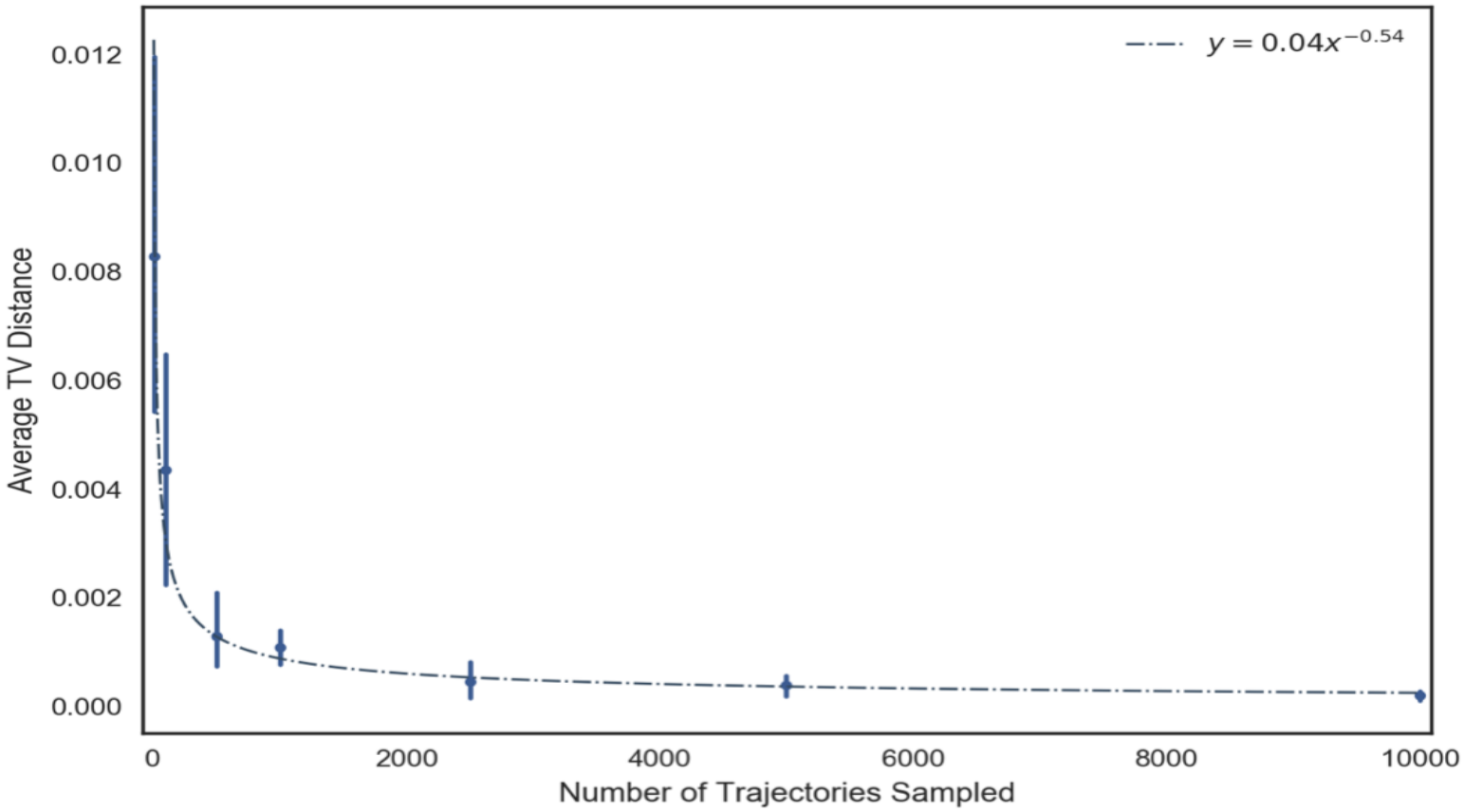
Sample Complexity – Qualitative Results

$$\|\pi_\theta(\cdot|x) - \pi(\cdot|x)\| \leq O\left(\frac{1}{\sqrt{n}}\right) + \|\pi_\theta(\cdot|x) - \pi_n(\cdot|x)\|$$

For each data point:

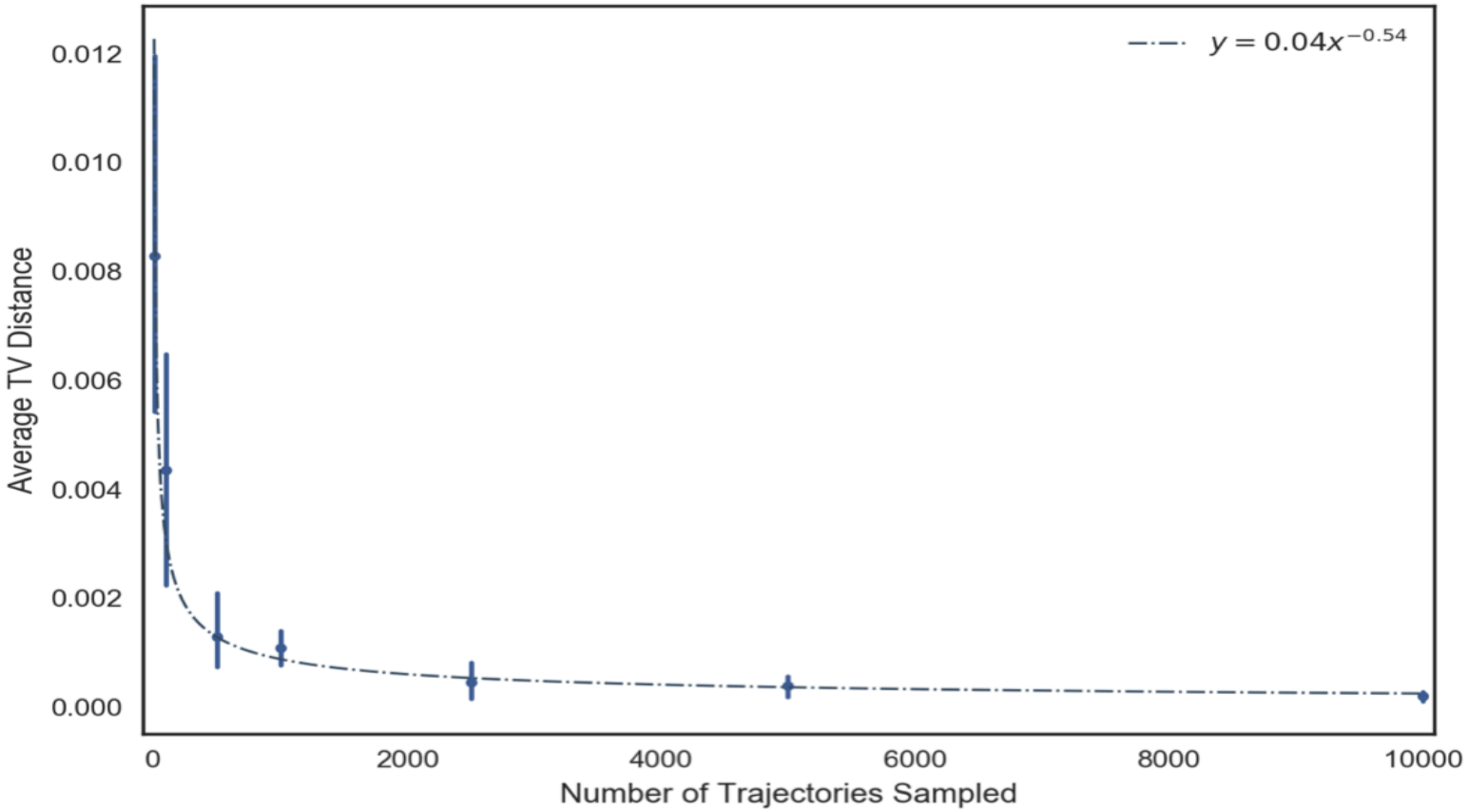
1. 5 datasets of the given size were collected.

2. For each dataset, the IRL was performed with 5 random initializations and the best performance on each dataset was recorded and averaged



Sample Complexity – Qualitative Results

This suggests the limiting factor in how well our IRL approach can perform, is how well the data approximates the true policy



$$\|\pi_\theta(\cdot|x) - \pi(\cdot|x)\| = O\left(\frac{1}{\sqrt{n}}\right)$$

Example 2: Ride Sharing

Sensitivity of agents to travel time and cost of ride-sharing trips

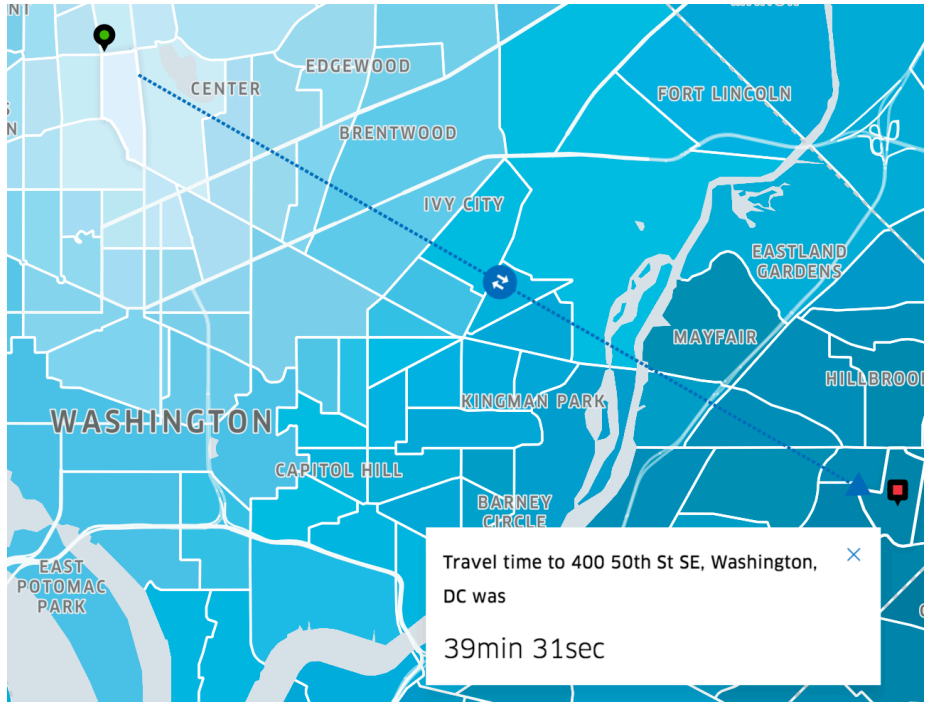
1. We collected data from a ride-sharing service and constructed an MDP, where the rewards had noise and were a mix of travel time and price:

    i. The states are price multiplier – time of departure tuples.

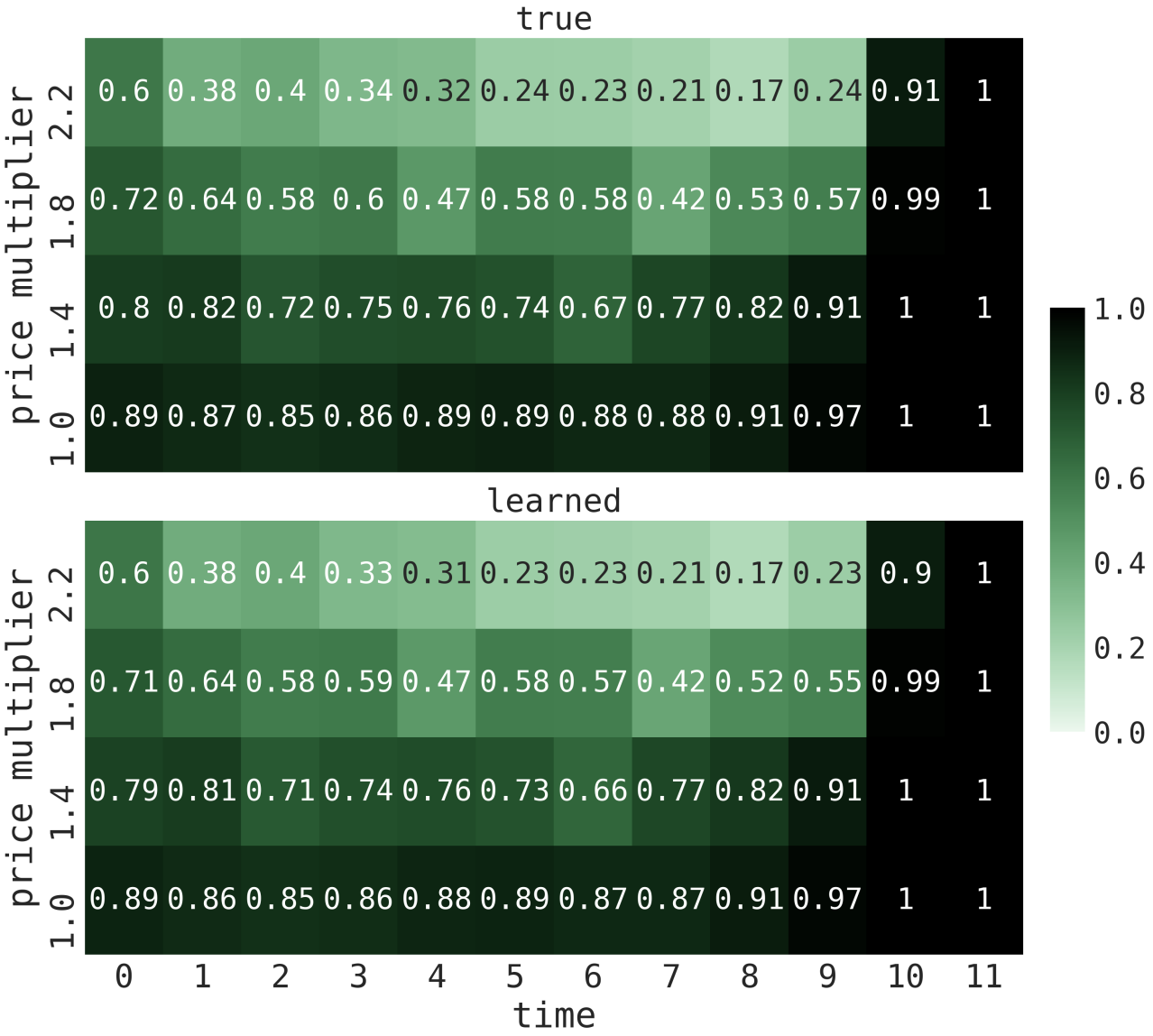
    ii. The actions are to either wait, or take a ride.

    iii. If the agent takes a ride, they exit the grid, otherwise they move to the next time step and the surge price increases or decreases with dynamics derived from data.

2. We then trained various agents in the MDP and performed the I-RSRL.



Example 2: Ride Sharing



- In each state we show the probability of taking the ride at the current surge price
- An agent that is risk-seeking on losses waits longer to take a ride if the surge is high

Again, the I-RSRL learns a value function that can accurately recreate the policy of the agent.

Summary

1. Overview of Risk-Sensitive Decision Making

2. Risk-Sensitive Q-Learning

3. Inverse Risk-Sensitive Reinforcement Learning
   - Problem Formulation
   - Theoretical Results
   - Example: Risk-Sensitive agents in Grid World
   - Example: Sensitivity to Surge-Pricing in Ride-Sharing

Conclusion and Future Work

- Design control and incentive schemes around the learned value functions to achieve some system goal
- Incorporate different risk measures (VaR, coherent risk measures, etc.)
- Learning representative utilities from populations of users
- Learning reference points
- Natural Gradients

References:

[1] A. Tversky and D. Kahneman, “Rational Choice and the Framing of Decisions,” J. Business, vol. 59, no. 4, pp. S251S278, 1986.

[2] D. Kahneman and A. Tversky, “Prospect theory: An analysis of decision under risk,” Econometrica, vol. 47, no. 2, pp. 263–291, 1979.

[3] Y. Shen, W. Stannat, and K. Obermayer, “Risk-Sensitive Markov Control Processes,” SIAM J. Control Optimization, vol. 51, 2013.

[4] S. I. Marcus, E. Fernandez-Gaucherand, D. Hernandez-Hernandez,. Coraluppi, and P. Fard, Risk Sensitive Markov Decision Processes. Birkhauser Boston, 1997.

[5] Y. Shen, M. J. Tobia, and K. Obermayer, “Risk-sensitive reinforcement learning,” Neural Computation, vol. 26, pp. 1298–1328, 2014.