535514: Reinforcement Learning

Lecture 27 — Inverse RL & Model-Based RL

Ping-Chun Hsieh

June 3, 2024

On-Policy vs Off-Policy Methods

<table>
<tr><th></th><th>Policy Optimization</th><th>Value-Based</th><th>Model-Based</th><th>Imitation-Based</th></tr>
<tr><td>On-Policy</td><td>Exact PG
REINFORCE (w/i baseline)
A2C
On-policy DAC
TRPO
Natural PG (NPG)
PPO-KL & PPO-Clip
RLHF by PPO-KL</td><td>Epsilon-Greedy MC
Sarsa
Expected Sarsa</td><td rowspan="2">MCTS
Model-Predictive Control (MPC)
PETS</td><td rowspan="2">IRL
GAIL
WAIL</td></tr>
<tr><td>Off-Policy</td><td>Off-policy DPG & DDPG
Twin Delayed DDPG (TD3)</td><td>Q-learning
Double Q-learning
DQN & DDQN
Rainbow
C51 / QR-DQN / IQN
Soft Actor-Critic (SAC)</td></tr>
</table>

Inverse RL: Occupancy Measure Matching

Brian Ziebart et al., Maximum entropy inverse reinforcement learning, AAAI 2008

Jonathan Ho and S. Ermon, Generative adversarial imitation learning, NIPS 2016

Xiao et al., Wasserstein Adversarial Imitation Learning, NeurIPS 2019

Garg et al., IQ-Learn: Inverse soft-Q Learning for Imitation, NeurIPS 2021

Review: Occupancy Measure Matching

$\pi(a|s)$

$P(a_t = a \mid s_t = s; \pi) \, P(s_t = s \mid s_0, \pi)$

Recall: *Occupancy measure* (or *discounted state-action visitation*)

$$d_\mu^\pi(s,a) := (1-\gamma)\mathbb{E}_{s_0\sim\mu}\left[\sum_{t=0}^{\infty}\gamma^t P(s_t = s, a_t = a \mid s_0, \pi)\right]$$

Property: $V^\pi(\mu) = \mathbb{E}_{(s,a)\sim d_\mu^\pi}[R(s,a)]$

reward function

$d_\mu^\pi(s,a) = \pi(a|s) \, d_\mu^\pi(s)$

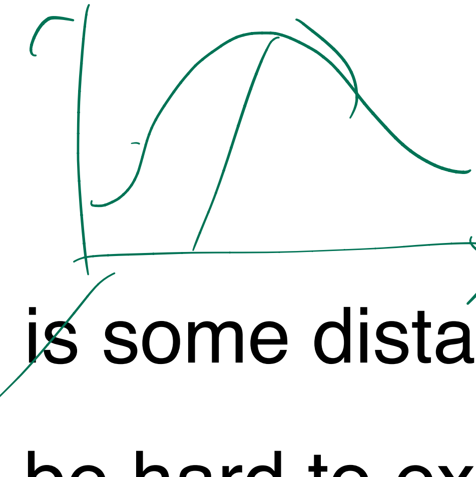
Occupancy measure matching:

expert policy

Find a policy π such that $d_\mu^\pi(s,a) = d^{\pi_e}(s,a), \quad \forall(s,a)$

Occupancy measure matching implies $V^\pi(\mu) = V^{\pi_e}(\mu)$

(Direct) Occupancy Measure Matching (OMM)



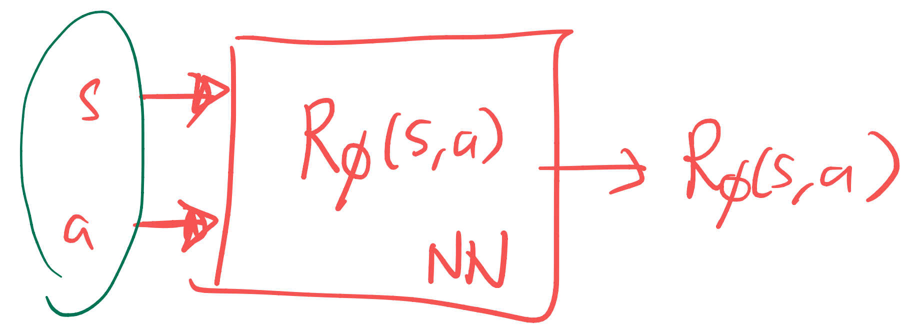
$$\min_{\pi\in\Pi} \quad L(\pi) := D(d_\mu^\pi, d_\mu^{\pi_e})$$

π_e is implicitly given (via an expert dataset)

($D(\cdot,\cdot)$ is some distance)

(d_μ^π could be hard to express!)

Dual of each other!



$$\max_{R\in\mathscr{R}} \min_{\pi\in\Pi} \underbrace{\left[\left(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^{\pi}}[R(s,a)]\right)\right]}_{:=L(\pi,R)}$$

OR

$$\min_{\pi\in\Pi} \max_{R\in\mathscr{R}} \underbrace{\left[\left(E_{(s,a)\sim d_\mu^{\pi_e}}[R(s,a)] - E_{(s,a)\sim d_\mu^{\pi}}[R(s,a)]\right)\right]}_{:=L(\pi,R)}$$

(Easier for training!)

To calculate $L(\pi,R)$:

① Draw (s,a) samples from expert dataset and also our policy π

② Do a forward pass to get $R(s,a)$

Apprenticeship Learning (APPLE)

A Motivating Example: Connecting OMM & APPLE

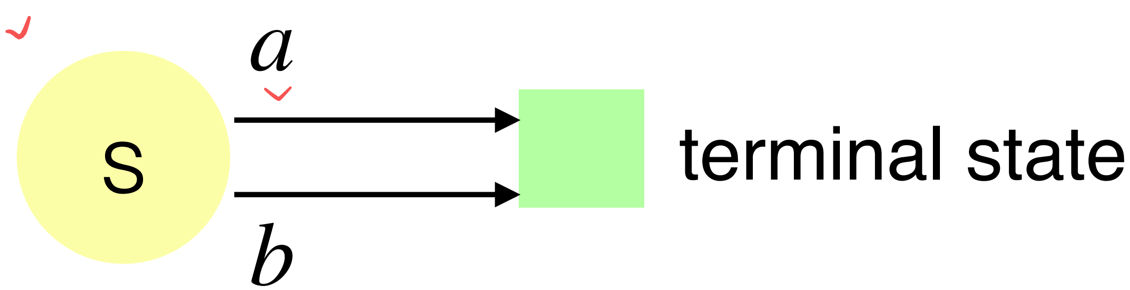
$\bar{L}(\pi)$

$$\min_{\pi\in\Pi} L(\pi) := D(d_\mu^\pi, d_\mu^{\pi_e}) \longleftrightarrow \cancel{\min_{\pi\in\Pi}}\max_{R\in\mathscr{R}} \underbrace{\left[\left(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^{\pi}}[R(s,a)]\right)\right]}_{:=L(\pi,R)}$$

$\pi_e(a|s)\cdot R(s,a) + \pi_e(b|s)\cdot R(s,a)$

$-\left((\pi(a|s)\cdot R(s,a) + \pi(b|s)\cdot R(s,b)\right))$

Consider a simple 1-state, 2-action MDP



Suppose $\mathscr{R} = \mathbb{R}^2$

expert policy: $\pi_e(a|s) = \pi_e(b|s) = 0.5$

Let's write down $R \in \mathscr{R}$ that maximizes $L(\pi, R)$ under a fixed π

For (s,a) with $d_\mu^\pi(s,a) > d_\mu^{\pi_e}(s,a)$: $R(s,a) = -\infty$

For (s,a) with $d_\mu^\pi(s,a) < d_\mu^{\pi_e}(s,a)$: $R(s,a) = +\infty$

For (s,a) with $d_\mu^\pi(s,a) = d_\mu^{\pi_e}(s,a)$: $R(s,a)$ can be anything

Case 1: For all (s,a), $d_\mu^\pi(s,a) = d_\mu^{\pi_e}(s,a)$

$\Rightarrow \bar{L}(\pi) = 0$

Case 2: There is at least one (s,a) with $d_\mu^\pi(s,a) \neq d_\mu^{\pi_e}(s,a)$

$\Rightarrow \bar{L}(\pi) = +\infty$

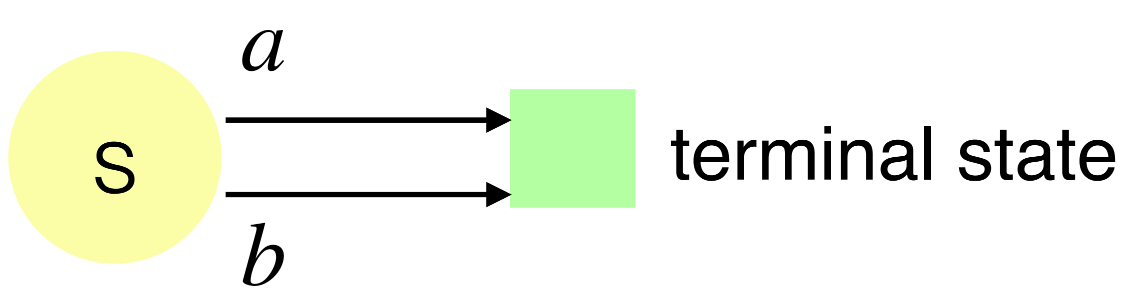
A Motivating Example: Connecting OMM & APPLE

$$\min_{\pi \in \Pi} L(\pi) := D(d_\mu^\pi, d_\mu^{\pi_e}) \longleftrightarrow \min_{\pi \in \Pi} \boxed{\max_{R \in \mathscr{R}} \Big[\underbrace{\Big(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^{\pi}}[R(s,a)] \Big)}_{:= L(\pi, R)} \Big]}$$

$\bar{L}(\pi)$

APPLE is equivalent to OMM with $\bar{L}(\pi) \equiv D$

Consider a simple 1-state, 2-action MDP



Suppose $\mathscr{R} = \mathbb{R}^2$

$$\pi_e(a|s) = \pi_e(b|s) = 0.5$$

Nice Property: Under $\mathscr{R} = \mathbb{R}^2$, the corresponding metric D is

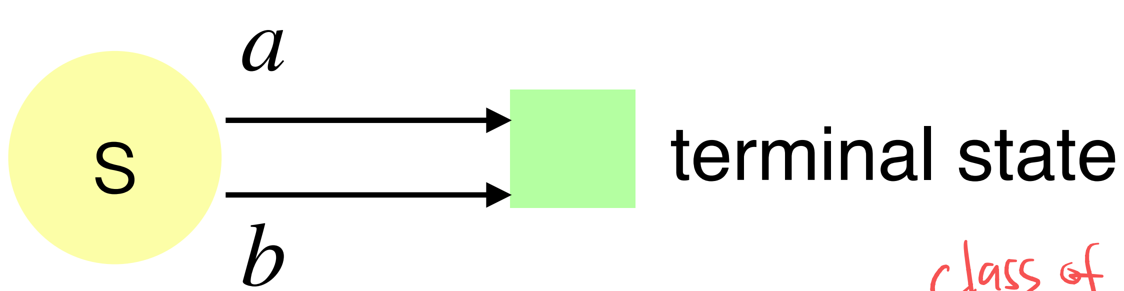
$$D(d_\mu^\pi, d_\mu^{\pi_e}) = \begin{cases} 0, & \text{if } d_\mu^\pi(s,a) = d_\mu^{\pi_e}(s,a), \forall (s,a) \\ \infty, & \text{otherwise} \end{cases}$$

A Motivating Example: Connecting OMM & APPLE (Cont.)

$$\min_{\pi\in\Pi} L(\pi) := D(d_\mu^\pi, d_\mu^{\pi_e}) \longleftrightarrow \min_{\pi\in\Pi}\max_{R\in\mathscr{R}} \underbrace{\left[\left(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^{\pi}}[R(s,a)]\right)\right]}_{:=L(\pi,R)}$$

$$= \bar{L}(\pi)$$

Consider a simple 1-state, 2-action MDP



Suppose $\mathscr{R} = \left\{ R \in \mathbb{R}^2 \,\middle|\, \|R\|_\infty \le 1 \right\}$

$|R(s,a)| \le 1$, for all (s,a)

class of bounded reward functions

$$\pi_e(a|s) = \pi_e(b|s) = 0.5$$

Let's write down $R \in \mathscr{R}$ that maximizes $L(\pi, R)$ under a fixed π

For (s,a) with $d_\mu^\pi(s,a) > d_\mu^{\pi_e}(s,a)$: $R(s,a) = -1$

For (s,a) with $d_\mu^\pi(s,a) < d_\mu^{\pi_e}(s,a)$: $R(s,a) = +1$

For (s,a) with $d_\mu^\pi(s,a) = d_\mu^{\pi_e}(s,a)$: $R(s,a)$ can be anything between 1 and -1

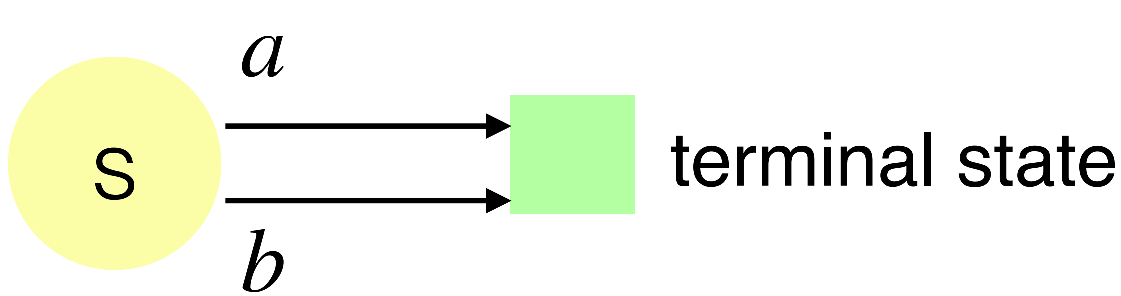
$$\bar{L}(\pi) = \sum_{(s,a)} \left(d_\mu^{\pi_e}(s,a) - d_\mu^\pi(s,a)\right) R(s,a) = \sum_{(s,a)} \left|d_\mu^{\pi_e}(s,a) - d_\mu^\pi(s,a)\right|$$

L1-norm (or total variation)

A Motivating Example: Connecting OMM & APPLE (Cont.)

$$\min_{\pi \in \Pi} \; L(\pi) := D(d_\mu^\pi, d_\mu^{\pi_e}) \longleftrightarrow \min_{\pi \in \Pi} \max_{R \in \mathscr{R}} \underbrace{\left[\left(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^{\pi}}[R(s,a)] \right) \right]}_{:= L(\pi, R)}$$

Consider a simple 1-state, 2-action MDP



Suppose $\mathscr{R} = \left\{ R \in \mathbb{R}^2 \,\middle|\, \|R\|_\infty \le 1 \right\}$

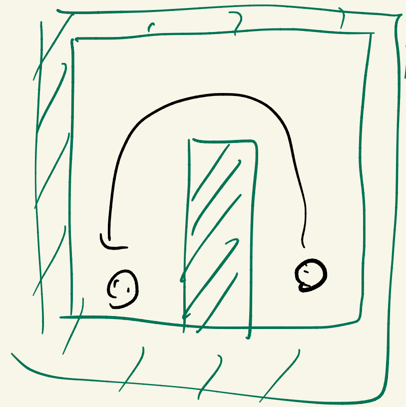
$$\pi_e(a|s) = \pi_e(b|s) = 0.5$$

Nice Property: Under $\mathscr{R} = \left\{ R \in \mathbb{R}^2 \,\middle|\, \|R\|_\infty \le 1 \right\}$, the metric D is

$$D(d_\mu^\pi, d_\mu^{\pi_e}) = \sum_{(s,a)} \left| d_\mu^\pi(s,a) - d_\mu^{\pi_e}(s,a) \right|$$

(usually called *"total variation distance"*)

How to choose $\mathscr{R}$ to get some widely-used D?



Example #1: Wasserstein Metric and APPLE

$$\min_{\pi \in \Pi} \; L(\pi) := W_p(d_\mu^\pi, d_\mu^{\pi_e}) \quad \longleftrightarrow \quad \min_{\pi \in \Pi} \max_{R \in \mathscr{R}} \underbrace{\left[\left(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^\pi}[R(s,a)] \right) \right]}_{:= L(\pi, R)}$$

(Wasserstein)

where $\mathscr{R} = \left\{ R \in \mathbb{R}^{\mathcal{S} \times \mathscr{A}} \,\middle|\, \text{Lip}(R) \leq 1 \right\}$

Lipschitz continuous :

This is also known as the *Kantorovich-Rubenstein duality*

A function f is Lip. conti.

if $|f(x) - f(y)| \leq C \cdot \|x-y\|_2$

Xiao et al., Wasserstein Adversarial Imitation Learning, NeurIPS 2019

Wasserstein Metric

F is the CDF of a random variable U

G " " " " V

Metric for random vectors

- $U : \Omega \to \mathbb{R}^d$: a random vector from the sample space Ω to $\mathbb{R}^d$
- For $1 \leq p < \infty$: $||U||_p := \left(\mathbb{E}\left[||U(\omega)||_p^p\right] \right)^{\frac{1}{p}}$

- **Wasserstein Metric**: For two CDFs F, G over the reals, the Wasserstein metric is defined as

$$W_p(F, G) := \inf_{\underbrace{(U,V)}_{\text{joint PDF}}:U \sim F, V \sim G} \underbrace{||U - V||_p}_{\text{cost}}$$

- Infimum is taken over <u>all joint distributions</u> of random variables (U, V), whose marginal distributions are F, G

Intuition Behind Wasserstein Metric

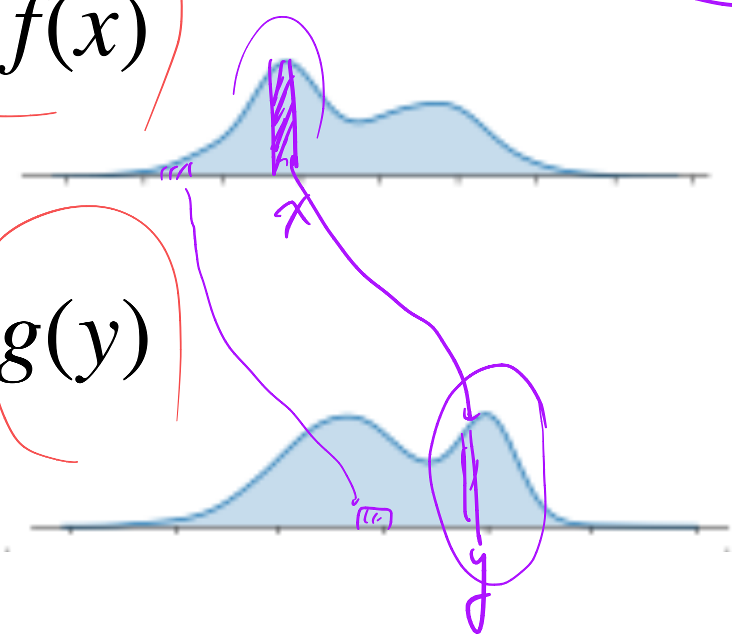
$f(x)$: marginal PDF of X
$g(y)$: " " of Y
$\alpha(x,y)$: joint PDF of X,Y

- Also known as: "optimal transport problem" or earth mover's distance
- Given two density $f(x), g(x)$ and a cost function $c(x, y)$ of moving mass from x to y, what is minimum cost of transforming from $f(x)$ to $g(y)$?

Total cost $= \int_{\text{all } x,y} \alpha(x,y) \cdot c(x,y)\, dx\, dy$

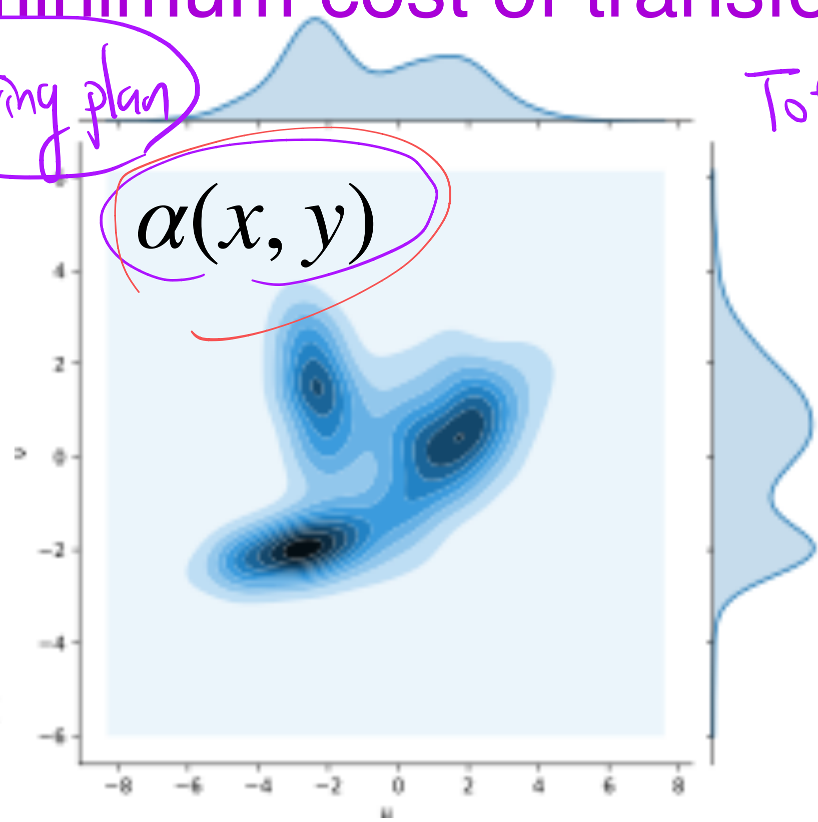
$f(x)$

$g(y)$



Moving plan

$\alpha(x, y)$



Minimum cost

$$C^* := \inf_{\alpha} \int c(x, y)\alpha(x, y)\,dx\,dy$$

optimal transport cost

$\alpha(x, y)$:amount of mass to move from x to y

$\alpha(x, y)$describes a feasible transport plan if

$$\int \alpha(x, y)dy = f(x), \quad \int \alpha(x, y)dx = g(y)$$

$\Rightarrow$ Conservation Law!

original amount at x

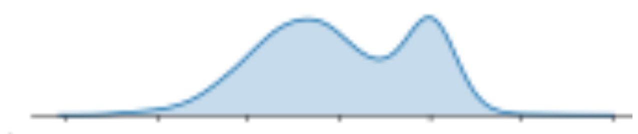
Summary: Optimal Transport & Wasserstein Metric

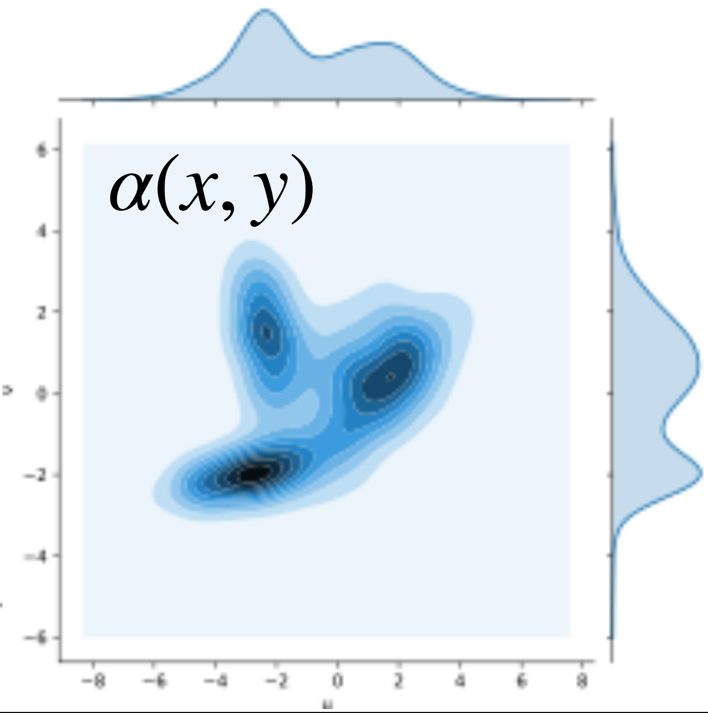
Wasserstein $W_p(F,G) := \inf_{(U,V):U\sim F, V\sim G} ||U-V||_p$

Optimal Transport (OT)

$f(x)$

$g(y)$





$c(x,y)$ = cost function of moving one unit of mass from x to y

- OT can be written as an optimization problem:

$$\min_{\alpha} \sum_{x,y} c(x,y)\alpha(x,y)$$

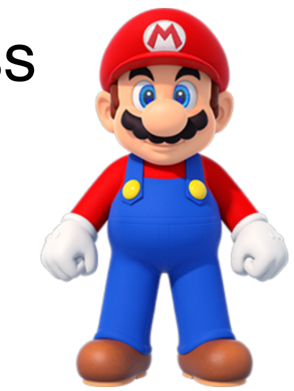
subject to (1) $\sum_{y} \alpha(x,y) = f(x), \forall x$ (2) $\sum_{x} \alpha(x,y) = g(y), \forall y$

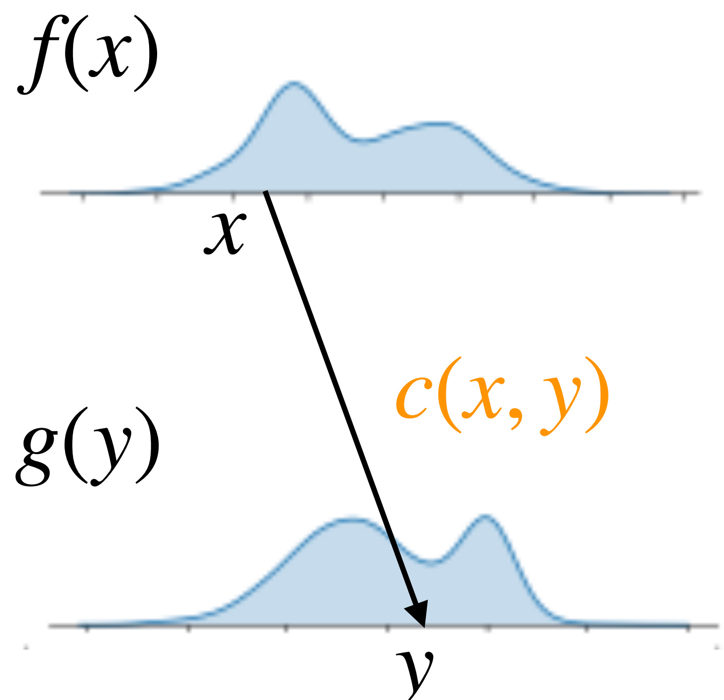
(3) $\alpha(x,y) \geq 0, \forall x, y$

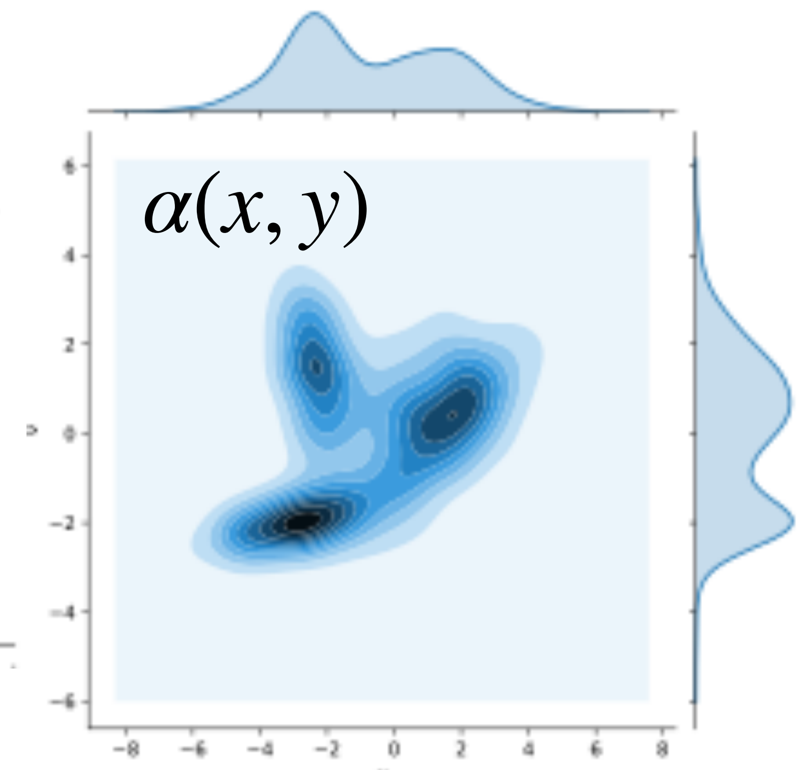
Duality of Optimal Transport: Economic Interpretation

Primal Form of OT (Mario moving the earth by himself)

$c(x, y)$ = Mario's cost function (for moving one unit of mass from x to y)

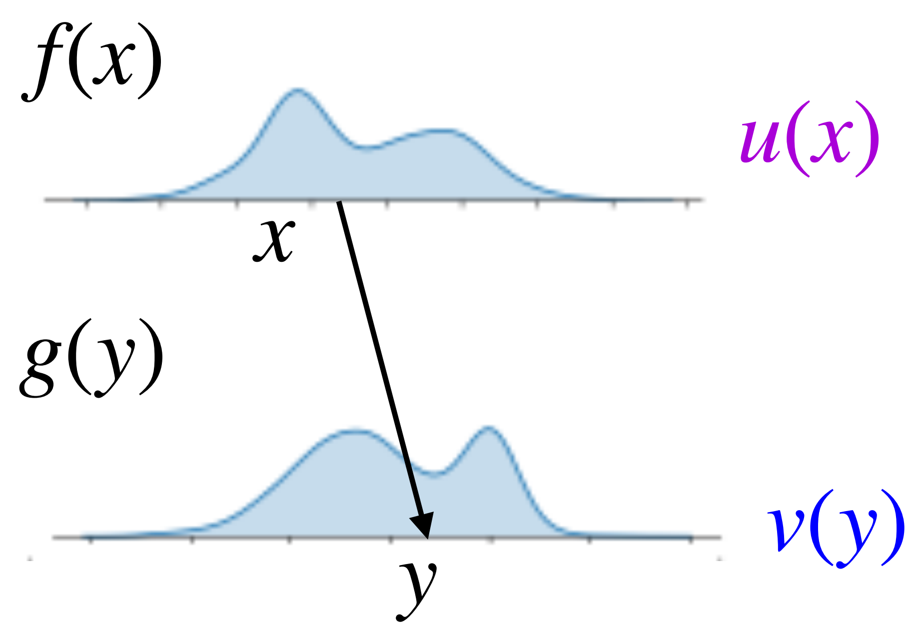






Dual Form of OT (Luigi offers to help Mario)

Mario needs to pay Luigi $u(x)$ and $v(y)$ (for moving one unit of mass from x to y)



Question: Under what condition would Mario ask for Luigi's help?

Duality of Optimal Transport (Formally)

- Primal Form of Optimal Transport

$$\min_{\alpha} \sum_{x,y} c(x,y)\alpha(x,y)$$

subject to (1) $\sum_{y} \alpha(x,y) = f(x), \forall x$ (2) $\sum_{x} \alpha(x,y) = g(y), \forall y$

(3) $\alpha(x,y) \geq 0, \forall x, y$

- Dual Form of Optimal Transport

Choose $f(x) \equiv d_{\mu}^{\pi_e}(s,a)$ | $u(x) \equiv R(s,a)$

$g(y) \equiv d_{\mu}^{\pi}(s,a)$ | $v(y) \equiv -R(s,a)$

$$\max_{u,v} \mathbb{E}_{x\sim f(x)}[u(x)] + \mathbb{E}_{y\sim g(y)}[v(y)]$$

subject to $u(x) + v(y) \leq c(x,y), \forall x, y$

The dual form looks exactly like APPLE!

- Both forms lead to the same optimal values (called "strong duality")

Example #2: Generative Adversarial Imitation Learning (GAIL)

- Recall: Dual Form of Optimal Transport

$$\max_{u,v} \mathbb{E}_{x\sim f(x)}[u(x)] + \mathbb{E}_{y\sim g(y)}[v(y)]$$

subject to $u(x) + v(y) \leq c(x,y), \forall x, y$

$s, a \rightarrow \boxed{\text{Binary Classifier } D_\phi(s,a)} \rightarrow D_\phi(s,a)$

$s \rightarrow \boxed{\text{Policy } \pi_\theta} \rightarrow$ action distribution

---

$D_\phi(s,a)$: A binary classifier that predicts the probability of the event that "the observed (s,a) is drawn from π"

Let's choose the following:

(1) $f(x) \equiv d_\mu^{\pi}(s,a)$

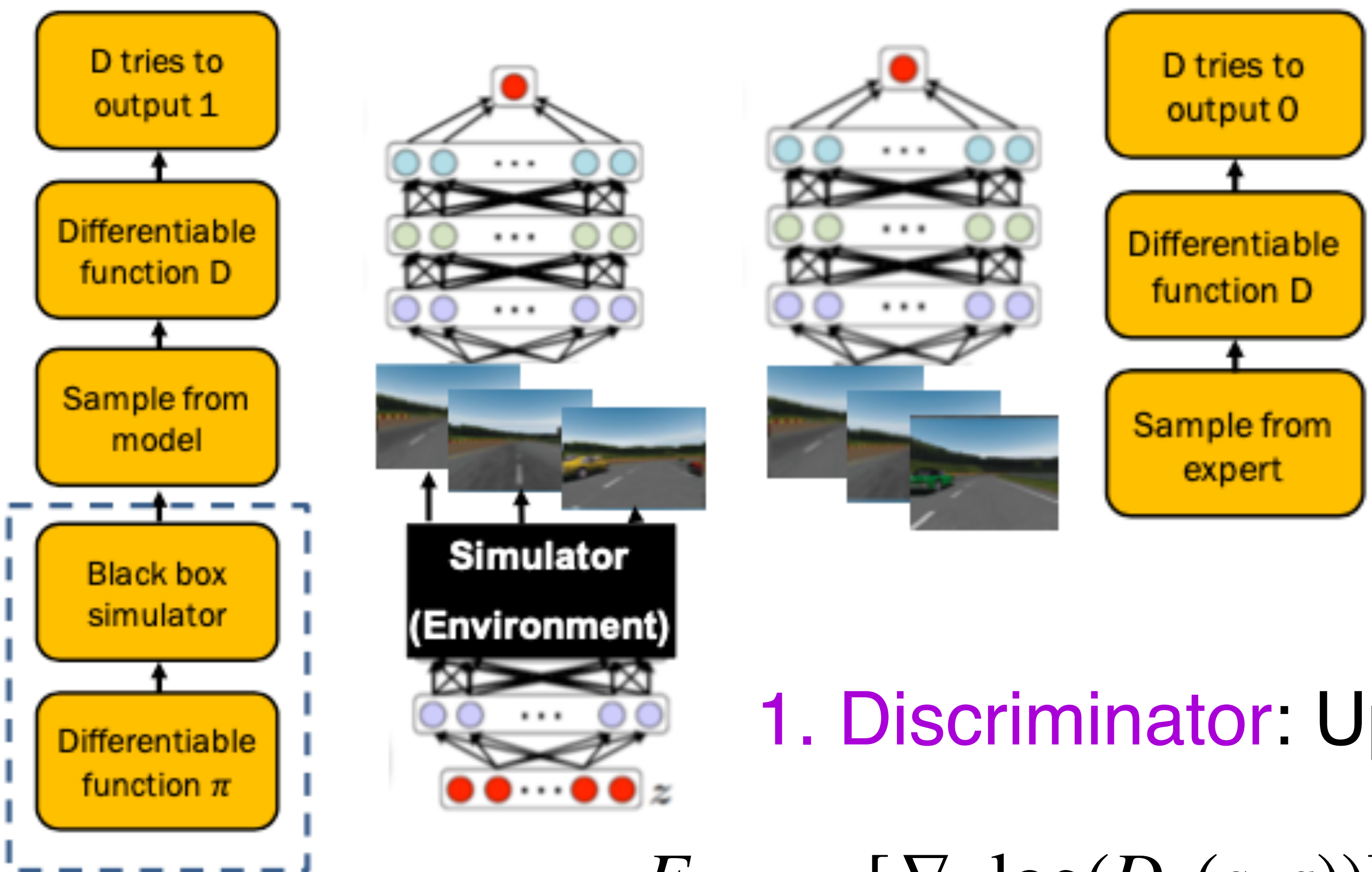
(2) $g(y) \equiv d_\mu^{\pi_e}(s,a)$

(3) $u(x) \equiv \log(D_\phi(s,a))$

(4) $v(y) \equiv \log(1 - D_\phi(s,a))$

→ Such choice eliminates the constraint!

GAIL: Discriminator and Generator



1. Discriminator: Update ϕ by

$$E_{(s,a)\sim d_\mu^{\pi}}[\nabla_\phi \log(D_\phi(s,a))] + E_{(s,a)\sim d_\mu^{\pi_e}}[\nabla_\phi \log(1-D_\phi(s,a))]$$

2. Generator: Use any RL algorithm with reward function $\log(D_\phi(s,a))$

Jonathan Ho and S. Ermon, Generative adversarial imitation learning, NIPS 2016

A Comparison Between Wasserstein AIL and GAIL

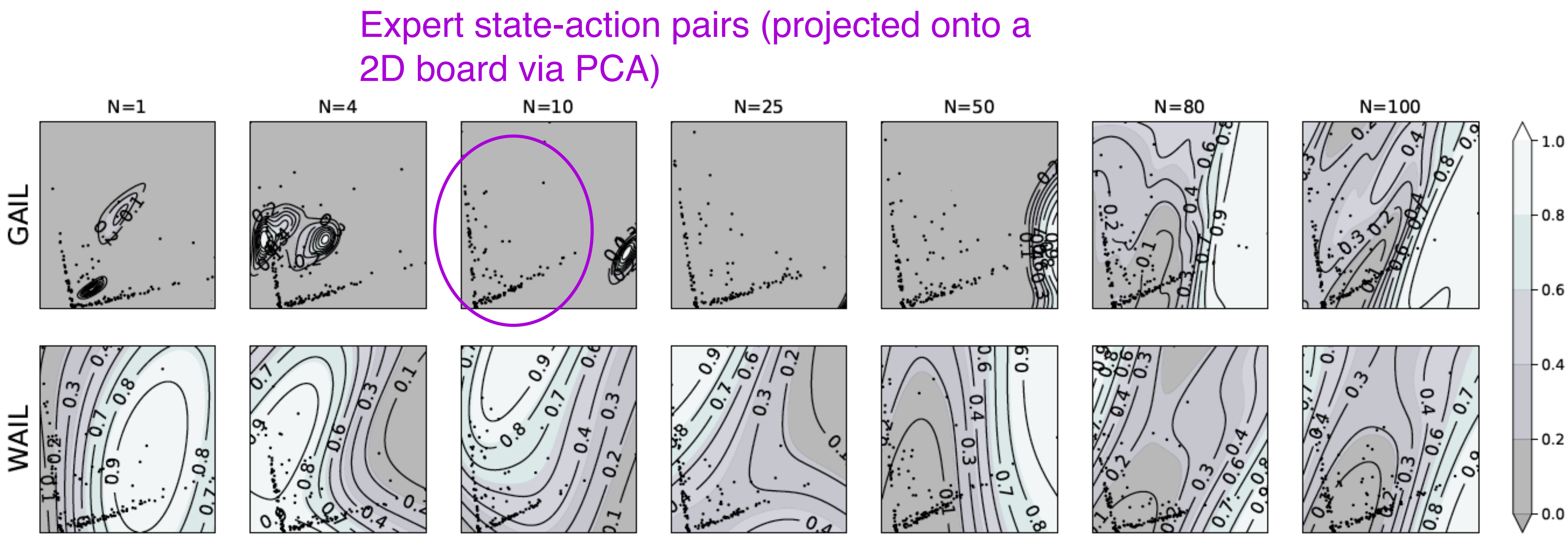


Figure 2: Reward surfaces of WAIL and GAIL on *Humanoid* with respect to different expert data sizes.

Xiao et al., Wasserstein Adversarial Imitation Learning, NeurIPS 2019

Summary: Occupancy Measure Matching via Apprenticeship Learning (With Regularization)

$$\min_{\pi \in \Pi} \max_{R \in \mathscr{R}} \underbrace{\left[\left(E_{(s,a)\sim d_\mu^{\pi_e}}[R(s,a)] - E_{(s,a)\sim d_\mu^{\pi}}[R(s,a)] \right) - H(\pi) + \psi(R) \right]}_{:=L(\pi,R)}$$

where $H(\pi) := E\left[\sum_t -\gamma^t \log \pi_t(a_t|s_t) \right]$ is the discounted causal entropy

$\psi(R)$ is a regularizer for the reward function

Key Idea: By choosing different "reward function classes $\mathscr{R}$", we obtain various OMM approaches!

π

V, Q

Model-Based RL

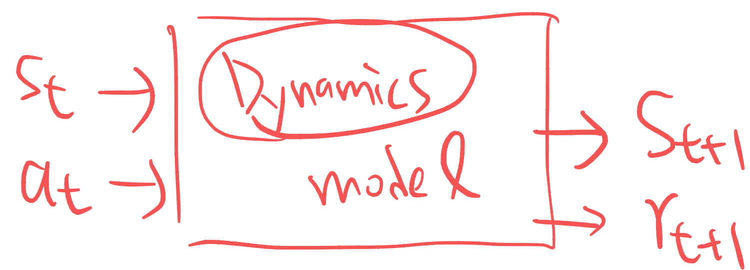
Moerland, Thomas M., et al. "Model-based reinforcement learning: A survey." *Foundations and Trends® in Machine Learning*

Part of the material is based upon the course material of CS285 by Sergey Levine

Model-Based Reinforcement Learning (MBRL)

1

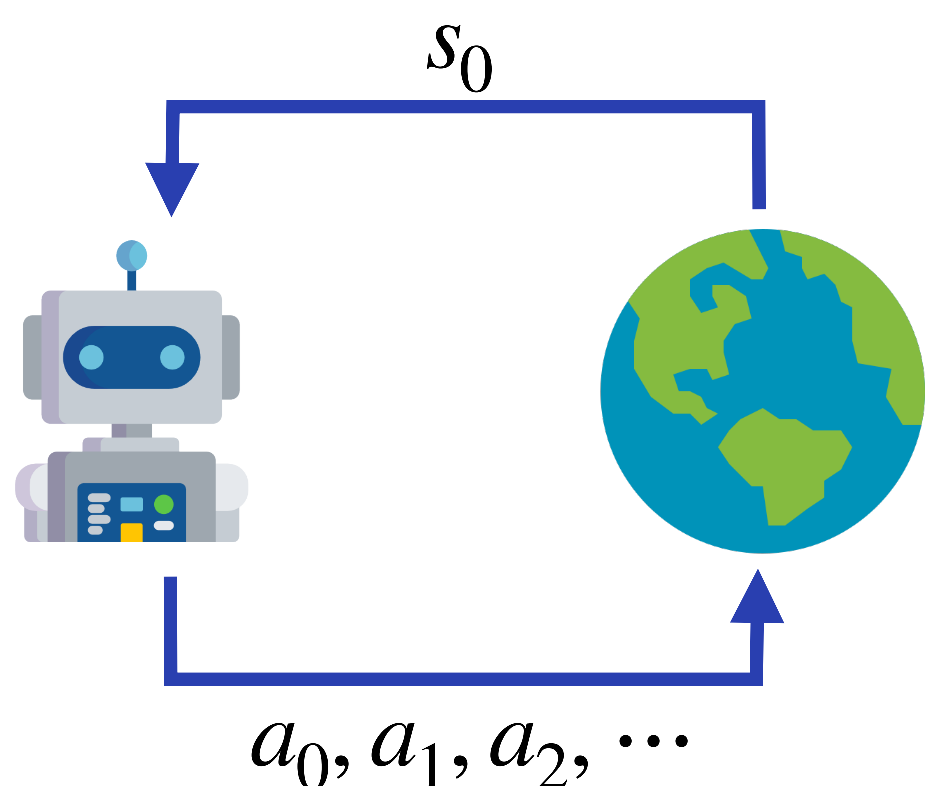
- Idea: Learn the dynamics model, and then determine the action sequence or a policy



- Today: How to determine action sequence if the dynamics model (P, R) is known
- Next Lecture: How to learn the dynamics model and apply MBRL in offline settings

The Simplest Case: Open-Loop Planning

- Open-loop planning: Given dynamics model and initial state s_0, determine the sequence of actions $a_0, a_1, a_2, \cdots, a_T$

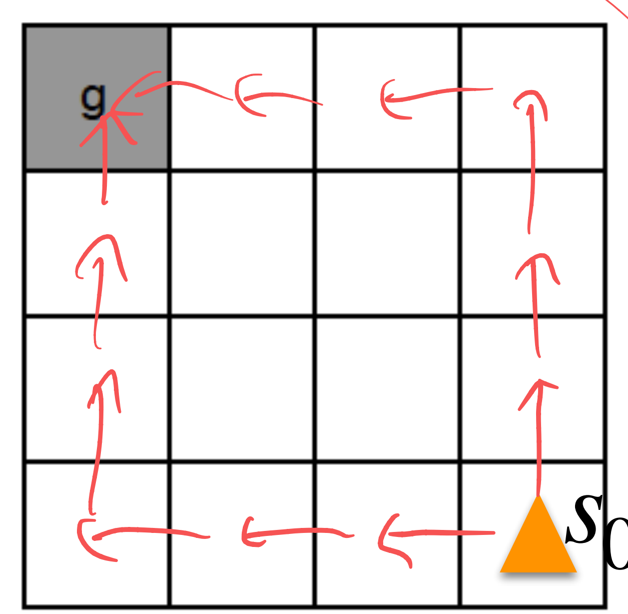


$$\arg\max_{a_0, a_1, \cdots, a_T} \mathbb{E}\left[\sum_{t=0}^{T} R(s_t, a_t)\right]$$

(For simplicity, let's assume $\gamma = 1$)

Example: Deterministic Gridworld

- Reward = -1, for each step
- Episode ends when reaching "g"



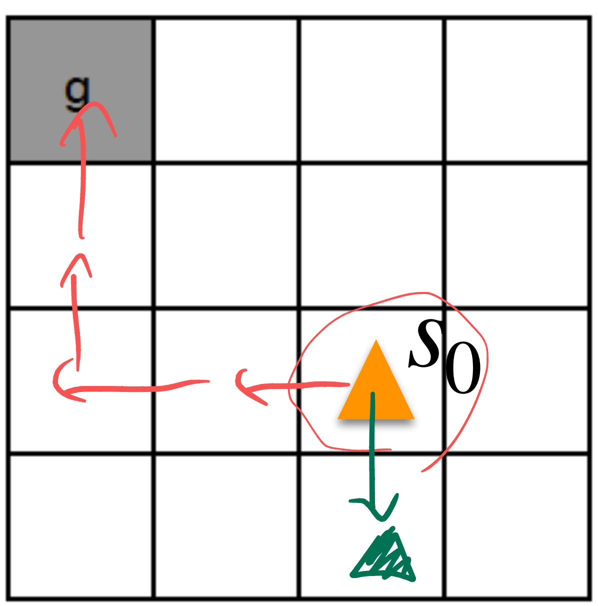
Open-Loop Planning in Stochastic Environments

Open-loop planning:

$$\arg\max_{a_0,a_1,\cdots,a_T} \mathbb{E}\left[\sum_{t=0}^{T} R(s_t, a_t)\right]$$

- However, open-loop planning can be sub-optimal in stochastic environments

Example: Stochastic Gridworld



Suppose 4 possible actions (Up, Down, Left, Right)

Suppose at each state:

(1) The next state follows the action, w.p. $1-\epsilon$

(2) The next state can be at any neighboring state, w.p. ϵ

(Suppose ϵ is small)

What's the selected action sequence under open-loop planning?

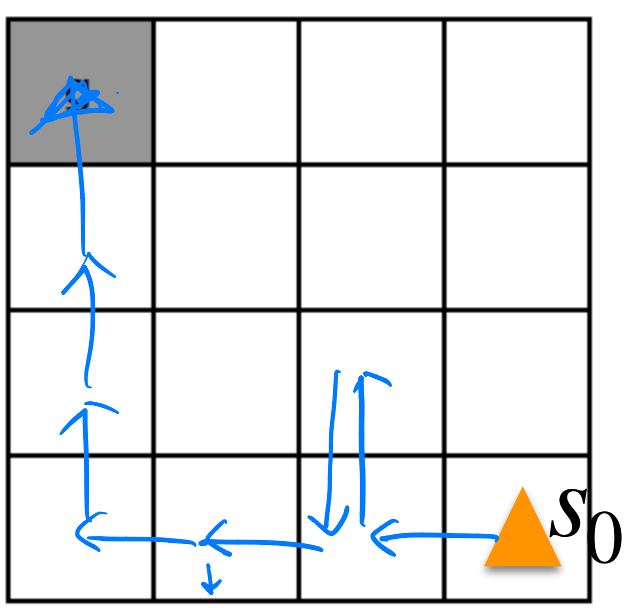
Open-Loop Planning by Stochastic Optimization

- Rethink open-loop planning as an optimization problem (this is often called "**trajectory optimization**")

$$\arg\max_{a_0, a_1, \cdots, a_T} \mathbb{E}\left[\sum_{t=0}^{T} R(s_t, a_t)\right] \xrightarrow{\theta := (a_0, \cdots, a_T)} \arg\max_{\theta} J(\theta)$$

(Here $J(\theta)$ is the total expected reward obtained under θ)

Example: Deterministic Gridworld



Suppose $T = 10$ and set θ as (L, U, D, L, D, L, U, U, U, D, R)

What's the corresponding $J(\theta)$?

$J(\theta) = -9$

Question: In general, given θ, how to obtain $J(\theta)$?

Solution 1: "Random Shooting"

- Rethink open-loop planning as an optimization problem (this is often called "**trajectory optimization**")

$$\arg\max_{a_0,a_1,\cdots,a_T} \mathbb{E}\left[\sum_{t=0}^{T} R(s_t, a_t)\right] \xrightarrow{\theta := (a_0, \cdots, a_T)} \arg\max_{\theta} J(\theta)$$

(Here $J(\theta)$ is the total expected reward obtained under θ)

- **Random shooting** (the simplest open-loop planning approach)

Step 1: Randomly draw action sequences $\theta^{(1)}, \theta^{(2)}, \cdots, \theta^{(K)}$ from some distribution (e.g., uniformly random)

Step 2: Choose $\theta^* = \arg\max_{k=1,\cdots,K} J(\theta^{(k)})$

Question: Is Random Shooting an efficient method? And why?

Solution 2: Cross-Entropy Method (CEM)

- **Cross-Entropy Method** (a popular open-loop planning approach)

Step 1: Randomly sample action sequences $\theta^{(1)}, \theta^{(2)}, \cdots, \theta^{(K)}$ from some distribution $\rho(\theta)$

Step 2: Evaluate $J(\theta^{(1)}), \cdots, J(\theta^{(K)})$ and choose $\theta^* = \arg\max_{k=1,\cdots,K} J(\theta^{(k)})$

Step 3: Select the top-N candidates $J(\theta^{(i_1)}), \cdots, J(\theta^{(i_N)})$

Step 4: Re-fit $\rho(\theta)$ to $\theta^{(i_1)}, \cdots, \theta^{(i_N)}$ by maximum likelihood estimation (MLE)

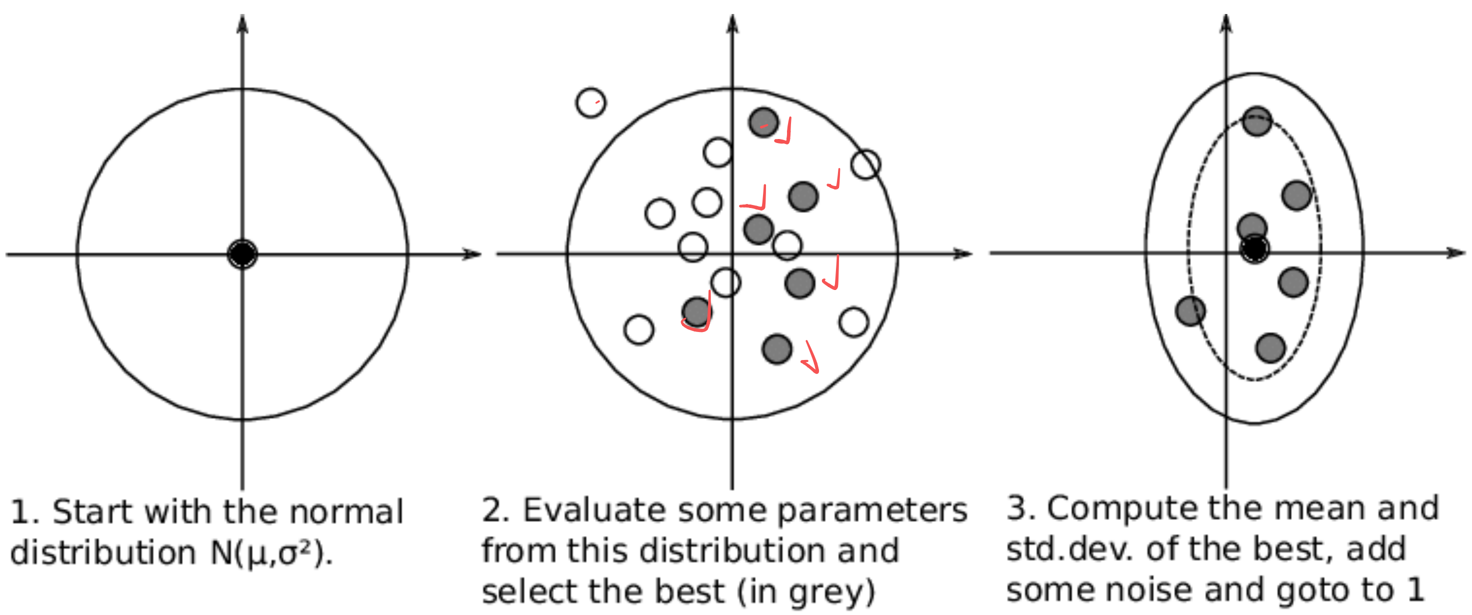


Figure Source: Towards fast and adaptive optimal control policies for robots: A direct policy search approach

Discussions on Open-Loop Planning

- Main advantages of open-loop planning:

(A1) Simple, zeroth-order optimization (no gradient at all!)

(A2) Very computationally efficient under parallelization

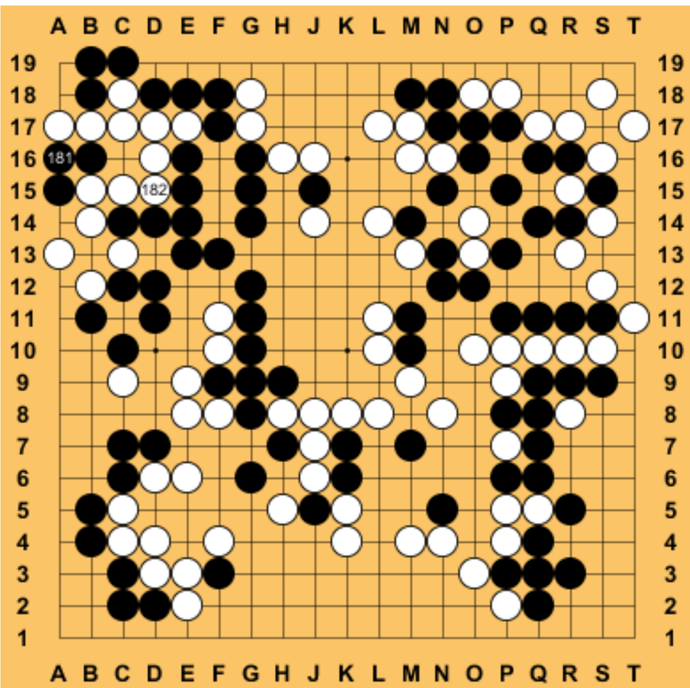
- Main drawbacks of open-loop planning:

(D1) Not scalable (dimensionally grows with T)

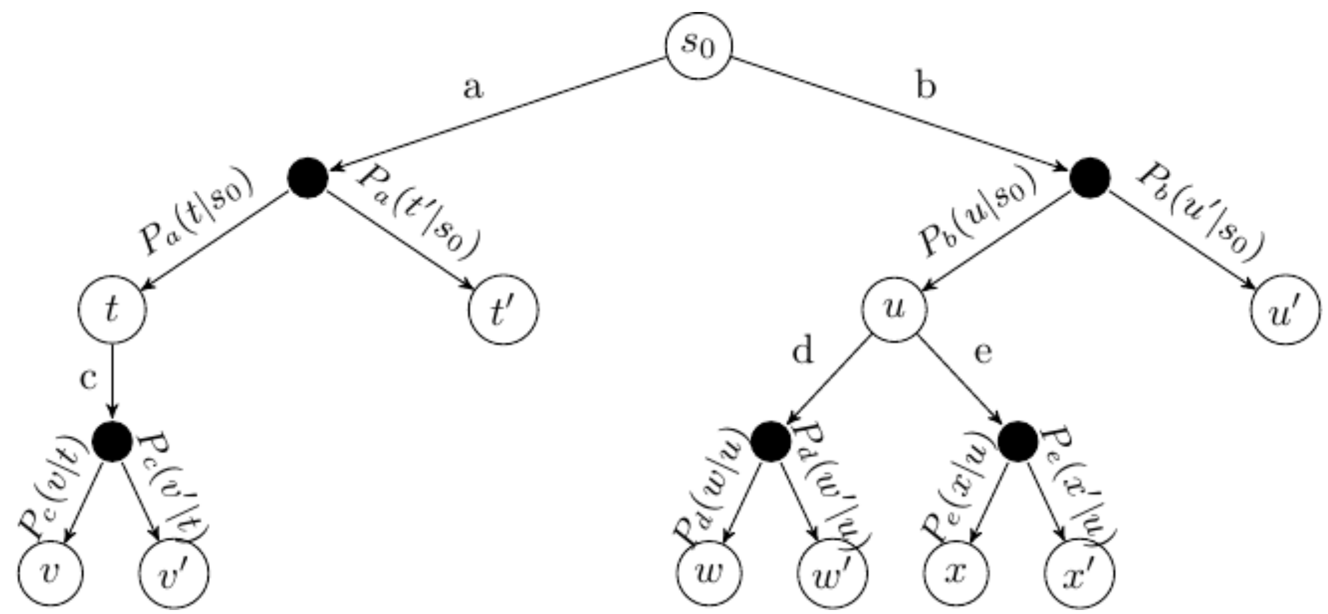
(D2) Can be sub-optimal (e.g., in stochastic environments)

Question: How to do more efficient "planning"?

For (D1): Monte-Carlo Tree Search (MCTS)



In discrete cases (state and action spaces are discrete), "θ" can be expressed as a tree



However, the number of nodes can grow exponentially with planning horizon T

Question: How to plan without building a full tree?

Intuition: Select the nodes with highest reward (exploit), but also spend some time visiting some unfamiliar nodes (explore)

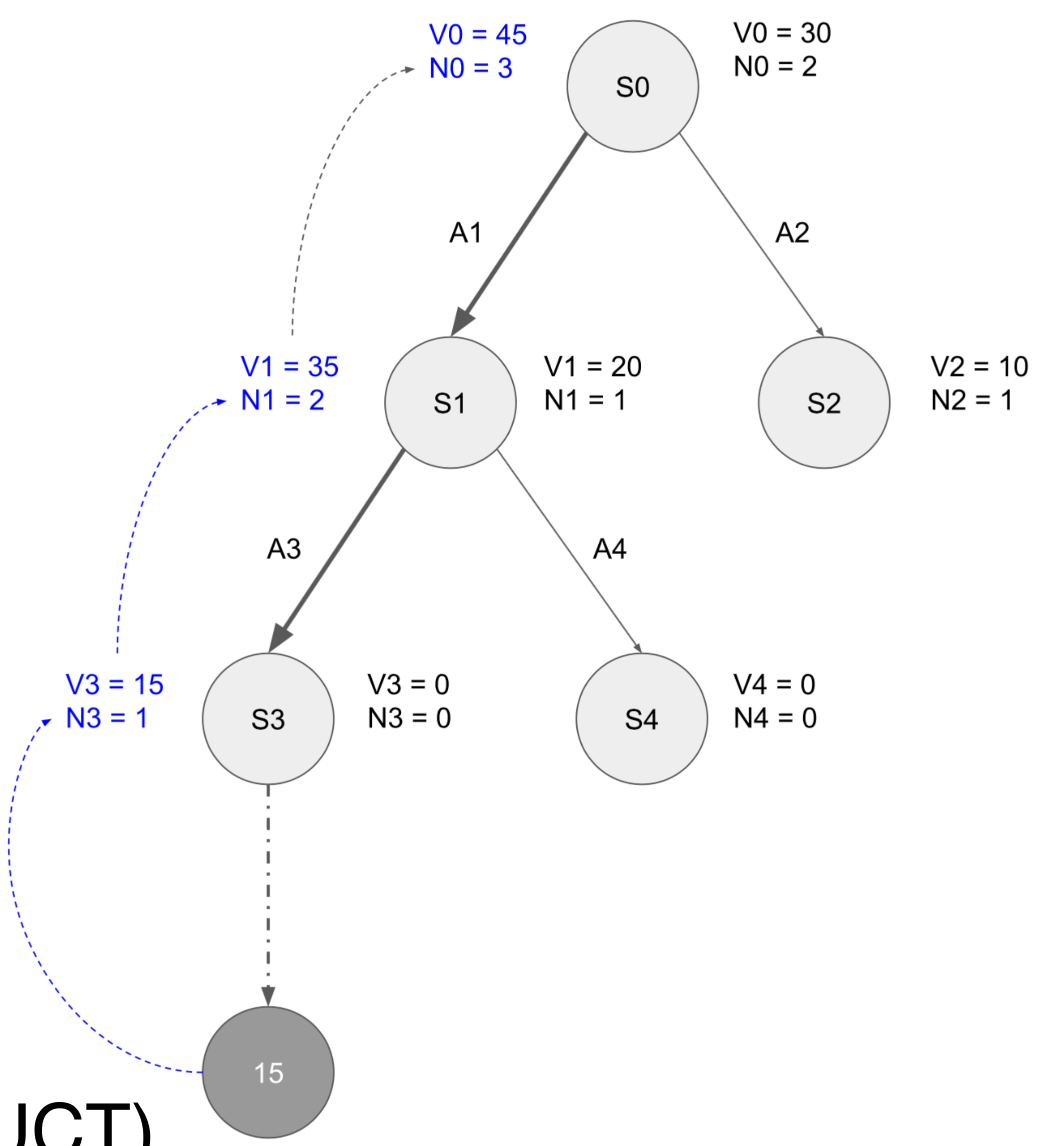
Case Study: Monte-Carlo Tree Search (MCTS)

- ## General Recipe of MCTS

Step 1: Traverse the tree from s_0 to a leaf node s_n by using TreePolicy

Step 2: Evaluate the leaf node s_n by some EvaluatePolicy

Step 3: Update the values of all the nodes between s_0 and s_n



A popular TreePolicy: Upper-Confidence Tree (UCT)

$$\text{Score of } (s, a) = \frac{V(s,a)}{N(s,a)} + C\sqrt{\frac{\log N(s)}{N(s,a)}}$$

Levente Kocsis and Csaba Szepesvari, "Bandit based Monte-Carlo Planning," ECML 2016

The Classic UCT Paper

Bandit based Monte-Carlo Planning

Levente Kocsis and Csaba Szepesvári

Computer and Automation Research Institute of the
Hungarian Academy of Sciences, Kende u. 13-17, 1111 Budapest, Hungary
`kocsis@sztaki.hu`

[ECML 2006]

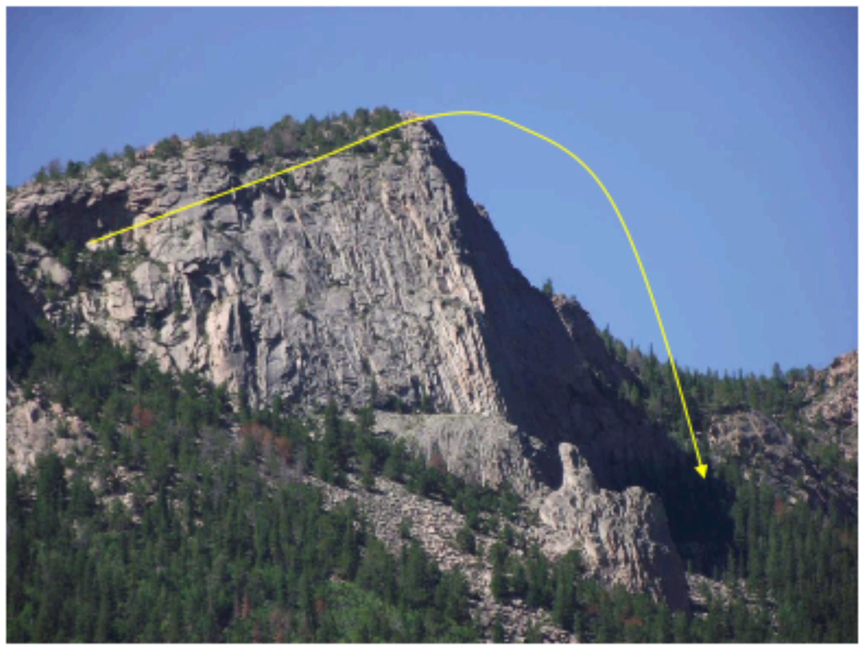
Abstract. For large state-space Markovian Decision Problems Monte-Carlo planning is one of the few viable approaches to find near-optimal solutions. In this paper we introduce a new algorithm, UCT, that applies bandit ideas to guide Monte-Carlo planning. In finite-horizon or discounted MDPs the algorithm is shown to be consistent and finite sample bounds are derived on the estimation error due to sampling. Experimental results show that in several domains, UCT is significantly more efficient than its alternatives.

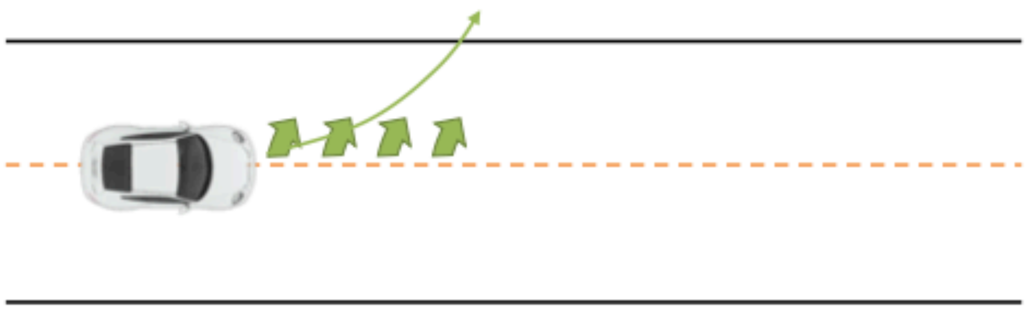
Levente Kocsis

Csaba Szepesvari

For (D2): Closed-Loop Planning

- Why is open-loop planning sub-optimal?





1. Stochastic transitions can put as in unexpected or even dangerous states (even if the dynamics model is fully known)

2. If the dynamics mode is NOT known, then we can make more mistakes

- Close-Loop Planning: Replan as you go (replan to fix the mistakes)

(Figure Source: Sergey Levine)

Closed-Loop Planning: Model-Predictive Control (MPC)

- **Model-Predictive Control**

At each time step $t = 0,1,2\cdots, T$

Step 1: Plan from current state s_t for a future horizon $H < T$ (e.g., by random shooting or CEM)

Step 2: Execute the first planned action and observe the next state s_{t+1}

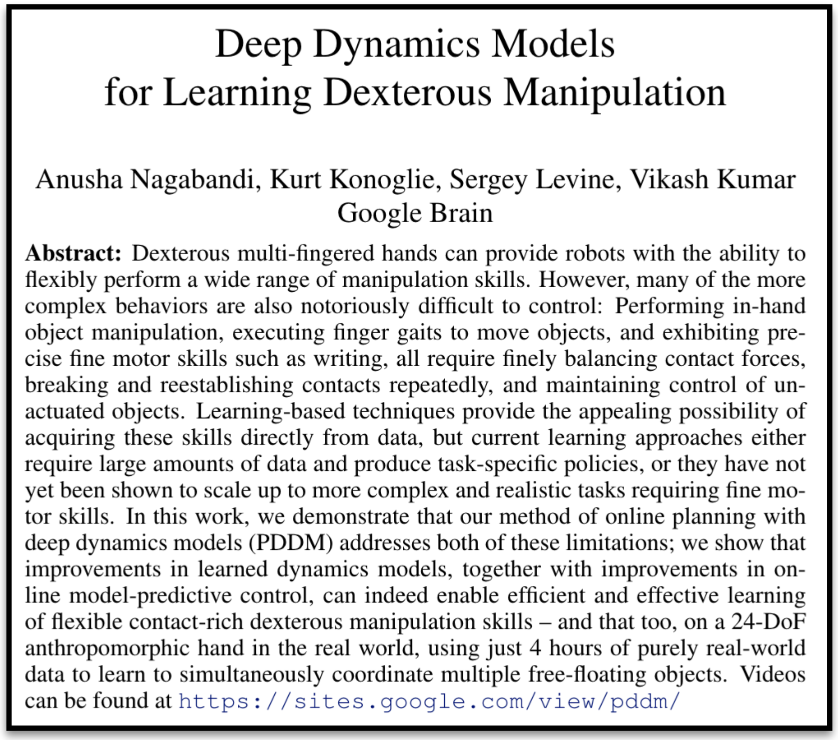
Nice features:

1. Short-horizon planning would work
2. Replanning helps with model errors
3. Even random shooting would work sufficiently well!

(Figure Source: Sergey Levine)

A Practical Example of MPC: PDDM for Robot Arm

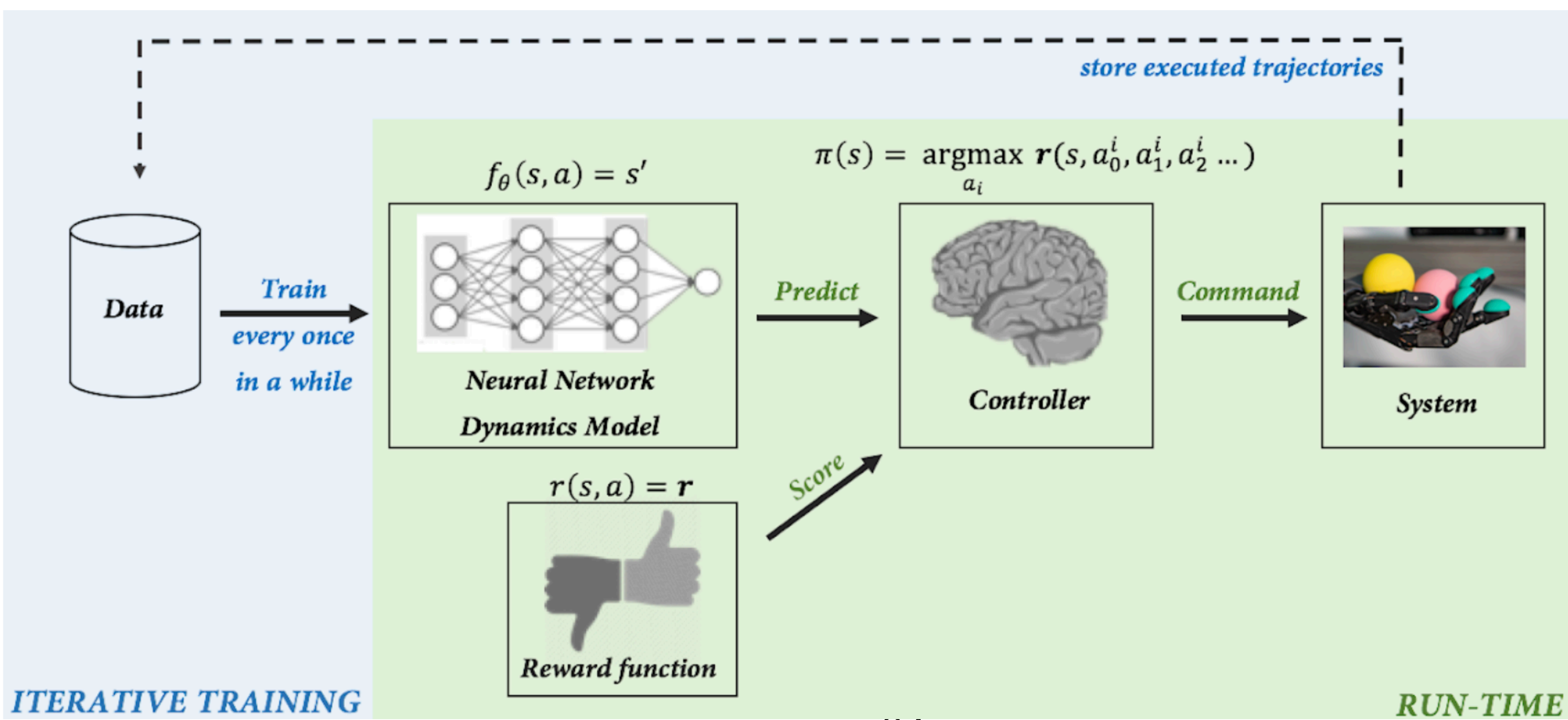
PDDM = Online **P**lanning with **D**eep **D**ynamics **M**odels



Deep Dynamics Models
for Learning Dexterous Manipulation

Anusha Nagabandi, Kurt Konoglie, Sergey Levine, Vikash Kumar
Google Brain

Abstract: Dexterous multi-fingered hands can provide robots with the ability to flexibly perform a wide range of manipulation skills. However, many of the more complex behaviors are also notoriously difficult to control: Performing in-hand object manipulation, executing finger gaits to move objects, and exhibiting precise fine motor skills such as writing, all require finely balancing contact forces, breaking and reestablishing contacts repeatedly, and maintaining control of unactuated objects. Learning-based techniques provide the appealing possibility of acquiring these skills directly from data, but current learning approaches either require large amounts of data and produce task-specific policies, or they have not yet been shown to scale up to more complex and realistic tasks requiring fine motor skills. In this work, we demonstrate that our method of online planning with deep dynamics models (PDDM) addresses both of these limitations; we show that improvements in learned dynamics models, together with improvements in online model-predictive control, can indeed enable efficient and effective learning of flexible contact-rich dexterous manipulation skills – and that too, on a 24-DoF anthropomorphic hand in the real world, using just 4 hours of purely real-world data to learn to simultaneously coordinate multiple free-floating objects. Videos can be found at https://sites.google.com/view/pddm/



https://sites.google.com/view/pddm/

Appendix: Proof of Kantorovich-Rubenstein Duality

Let's Show Kantorovich-Rubenstein Duality

$$\min_{\pi \in \Pi} L(\pi) := W(d_\mu^\pi, d_\mu^{\pi_e})$$

(Wasserstein)

$$\longleftrightarrow$$

$$\min_{\pi \in \Pi} \max_{R \in \mathscr{R}} \underbrace{\left[\left(E_{d_\mu^{\pi_e}}[R(s,a)] - E_{d_\mu^{\pi}}[R(s,a)] \right) \right]}_{:= L(\pi, R)}$$

where $\mathscr{R} = \left\{ R \in \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \,\middle|\, \text{Lip}(R) \leq 1 \right\}$

- Proof: $W(d_\mu^\pi, d_\mu^{\pi_e}) = \sup_{\mu(x)+\nu(y) \leq d(x,y)} \mathbb{E}_{x \sim f(x)}[\mu(x)] + \mathbb{E}_{y \sim g(\nu)}[\nu(x)]$

For any ν, one can push the term higher by setting $\mu(x) = \inf_y \{d(x,y) - \nu(y)\}$, and this implies $\mu(x) - \mu(y) \leq d(x,y)$, for all x, y

$$\begin{aligned}
W(d_\mu^\pi, d_\mu^{\pi_e}) &= \sup_{\nu} \sup_{\mu:\mu(x)+\nu(y) \leq d(x,y)} \mathbb{E}_{x \sim f(x)}[\mu(x)] + \mathbb{E}_{y \sim g(\nu)}[\nu(x)] \\
&= \sup_{\nu} \sup_{\mu: Lip(\mu) \leq 1, \mu(x)+\nu(y) \leq d(x,y)} \mathbb{E}_{x \sim f(x)}[\mu(x)] + \mathbb{E}_{y \sim g(\nu)}[\nu(x)] \\
&= \sup_{\mu: Lip(\mu) \leq 1} \sup_{\nu: \mu(x)+\nu(y) \leq d(x,y)} \mathbb{E}_{x \sim f(x)}[\mu(x)] + \mathbb{E}_{y \sim g(\nu)}[\nu(x)]
\end{aligned}$$

Finally, for any μ with $Lip(\mu) \leq 1$, ν is optimized at $\nu(y) = \inf_x \{d(x,y) - \mu(x)\}$

Since $Lip(\mu) \leq 1$, we have $\nu(y) = -\mu(y)$