Reward-Free RL is No Harder Than Reward-Aware RL in Linear Markov Decision Processes

Andrew Wagenmaker[1], Yifang Chen[1], Max Simchowitz[2], Simon S. Du[1], Kevin Jamieson[1]

1. University of Washington, 2. MIT

Reward-Aware vs Reward-Free RL

Reward-Aware vs Reward-Free RL

In the **reward-aware (PAC)** RL setting, the agent has access to the reward throughout exploration

Reward-Aware vs Reward-Free RL

In the **reward-aware (PAC)** RL setting, the agent has access to the reward throughout exploration

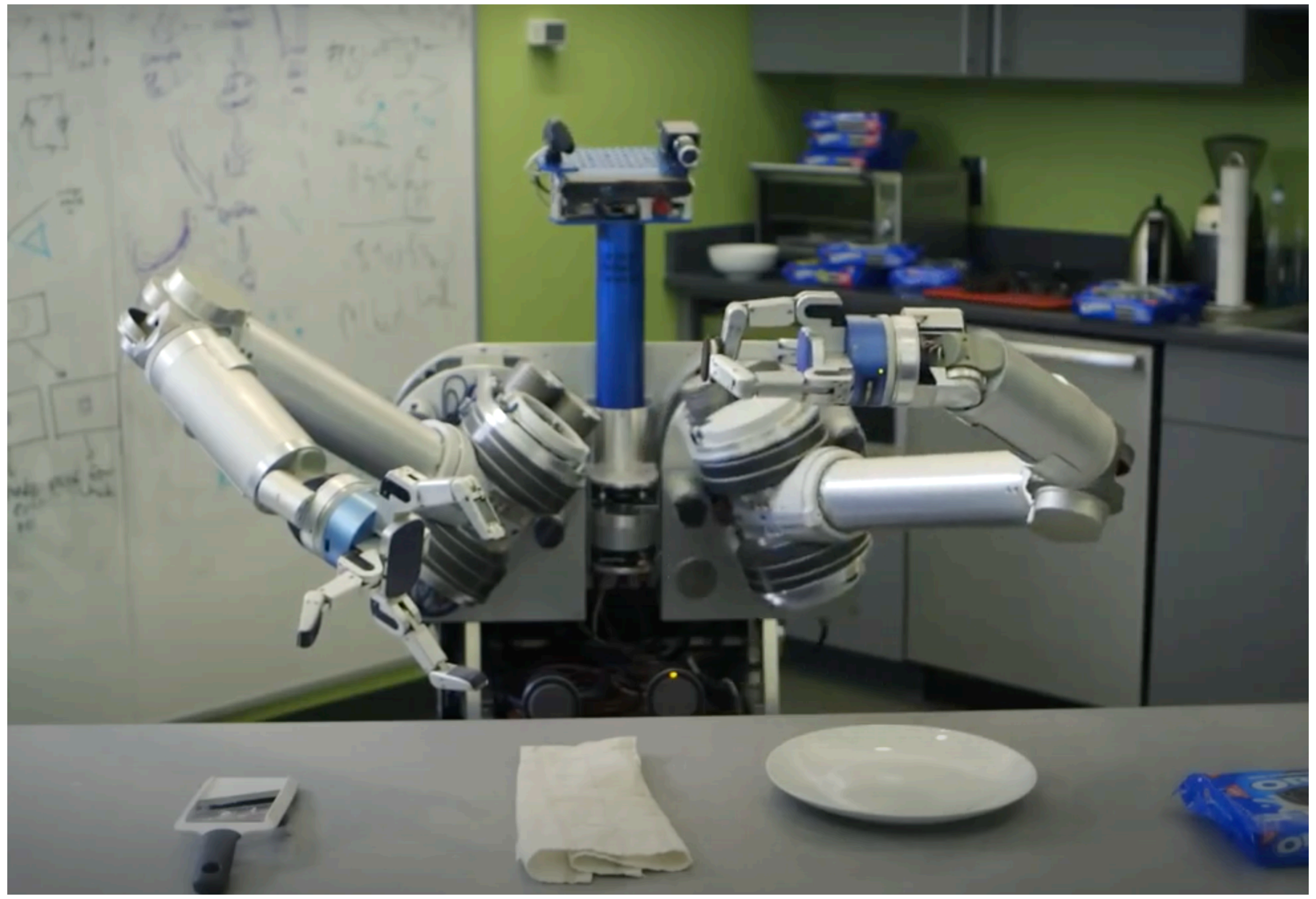
Reward-Aware vs Reward-Free RL

In the **reward-aware (PAC)** RL setting, the agent has access to the reward throughout exploration

In the **reward-free** setting, the agent is only given the reward after exploring

Reward-Aware vs Reward-Free RL

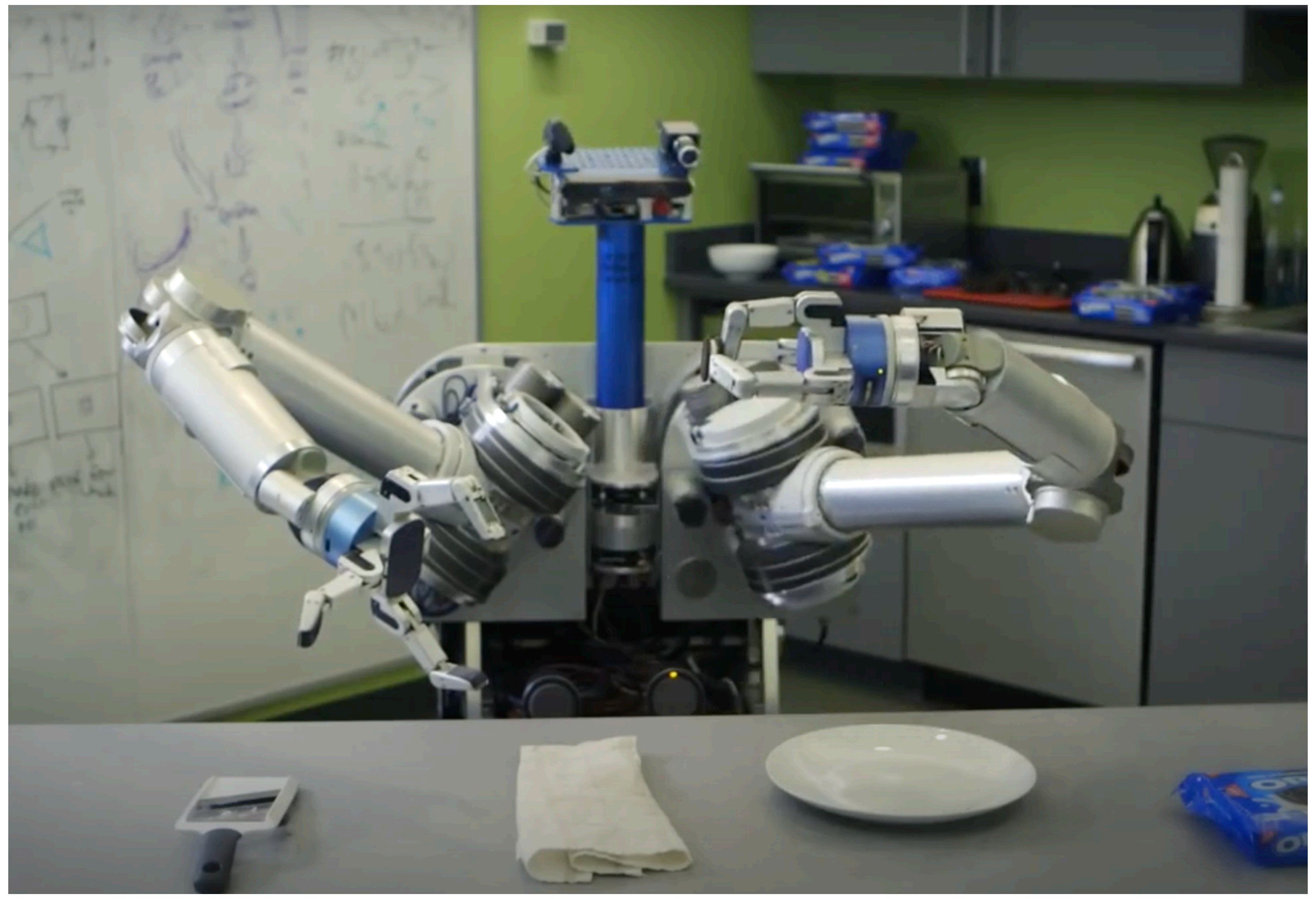
In the **reward-aware (PAC)** RL setting, the agent has access to the reward throughout exploration



In the **reward-free** setting, the agent is only given the reward after exploring

Motivation

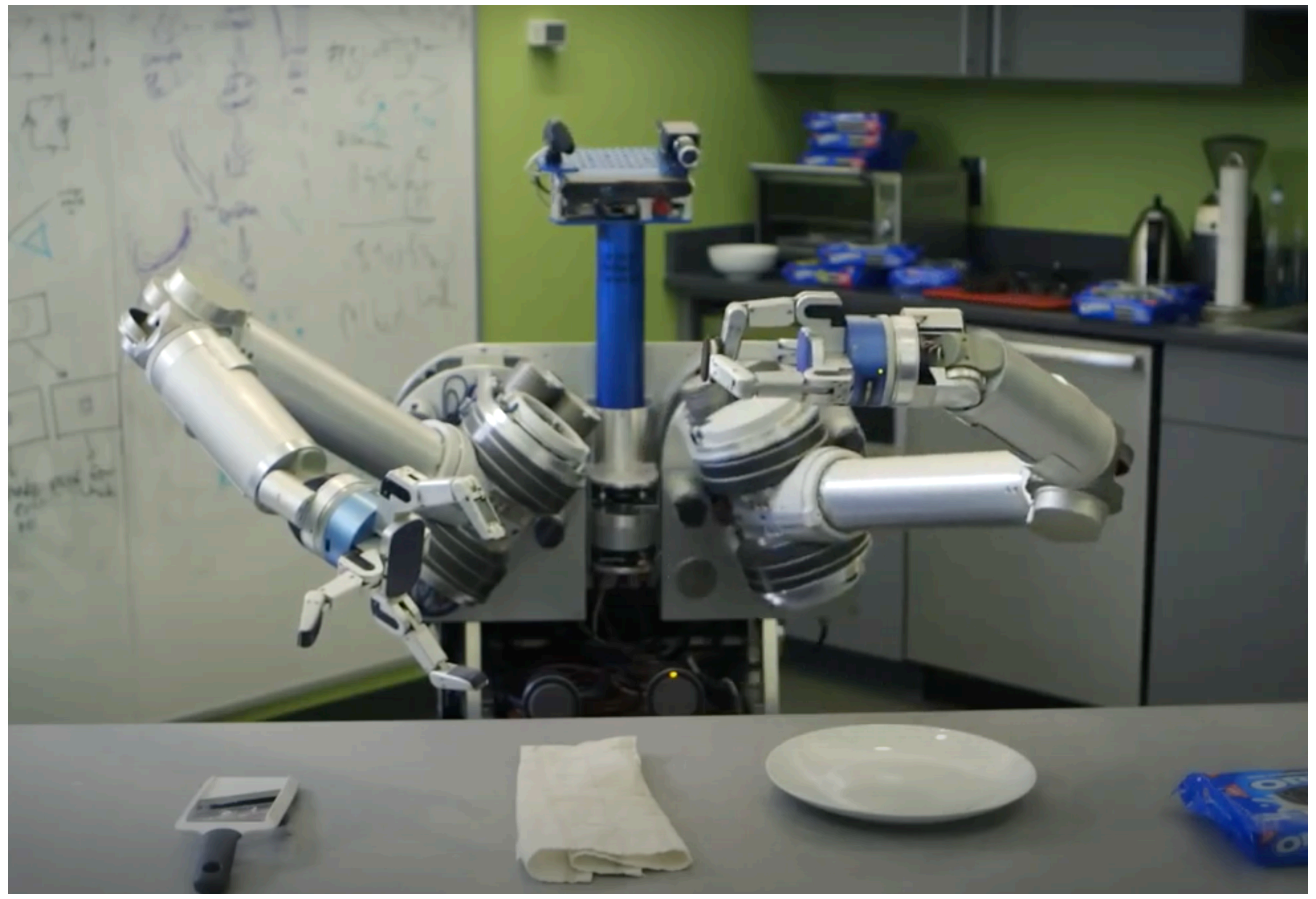
Intuitively, we would expect reward-free RL to be harder than reward-aware RL



Motivation

Intuitively, we would expect reward-free RL to be harder than reward-aware RL

In **tabular** MDPs:

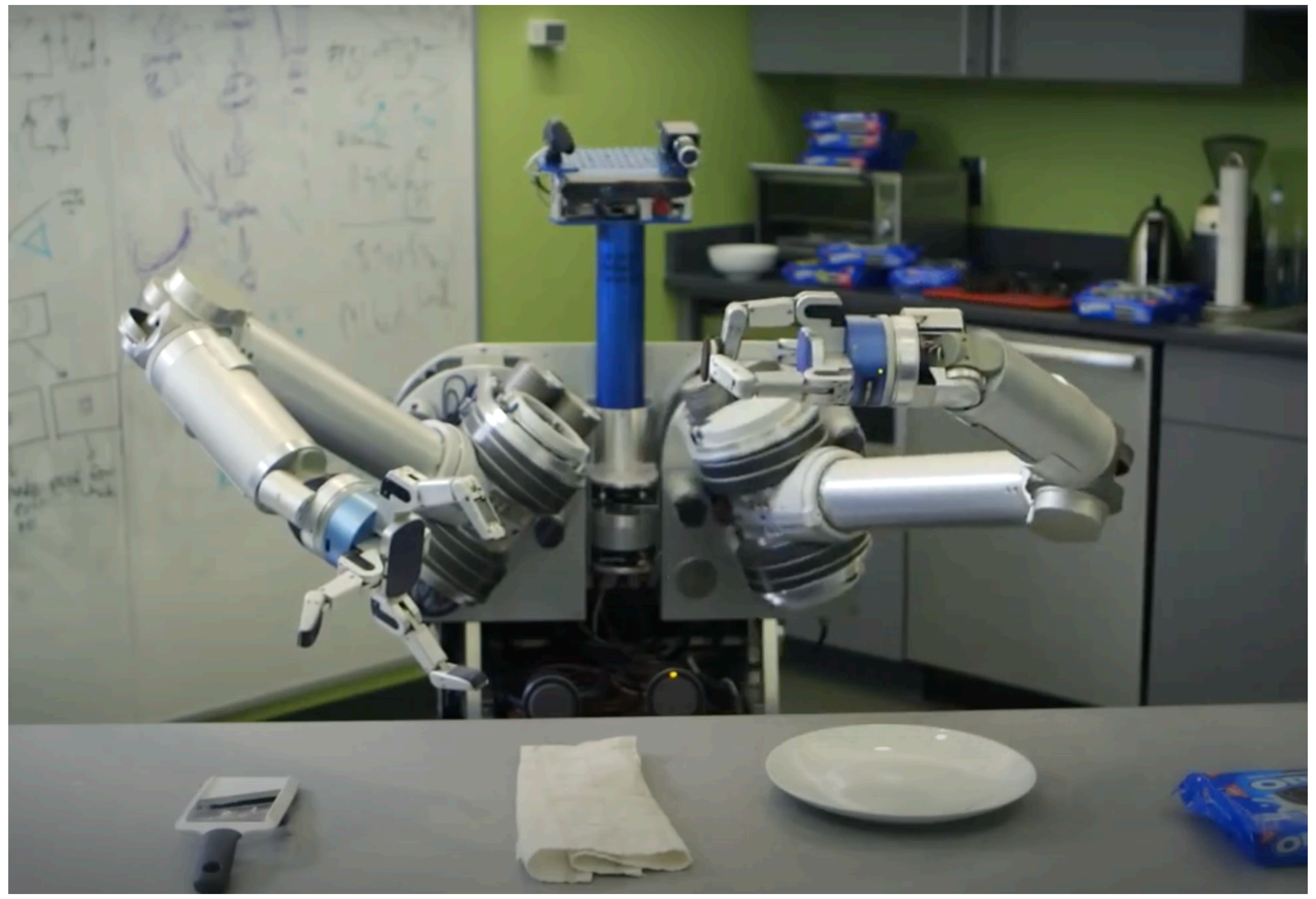


Motivation

Intuitively, we would expect reward-free RL to be harder than reward-aware RL

In **tabular** MDPs:

Optimal rate for **reward-aware** RL = $\Theta\left(\frac{SA}{\epsilon^2}\right)$



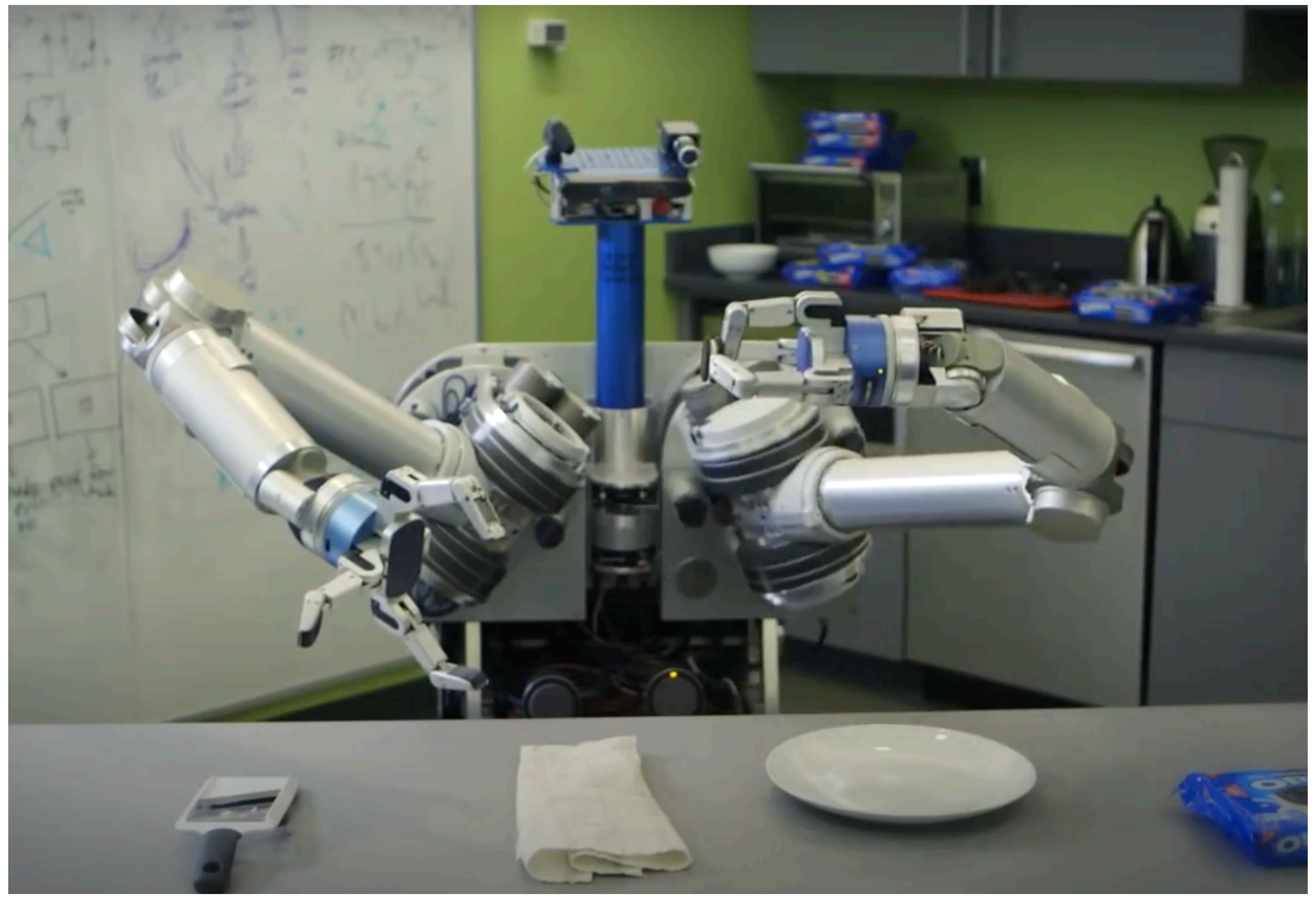
Motivation

Intuitively, we would expect reward-free RL to be harder than reward-aware RL

In **tabular** MDPs:

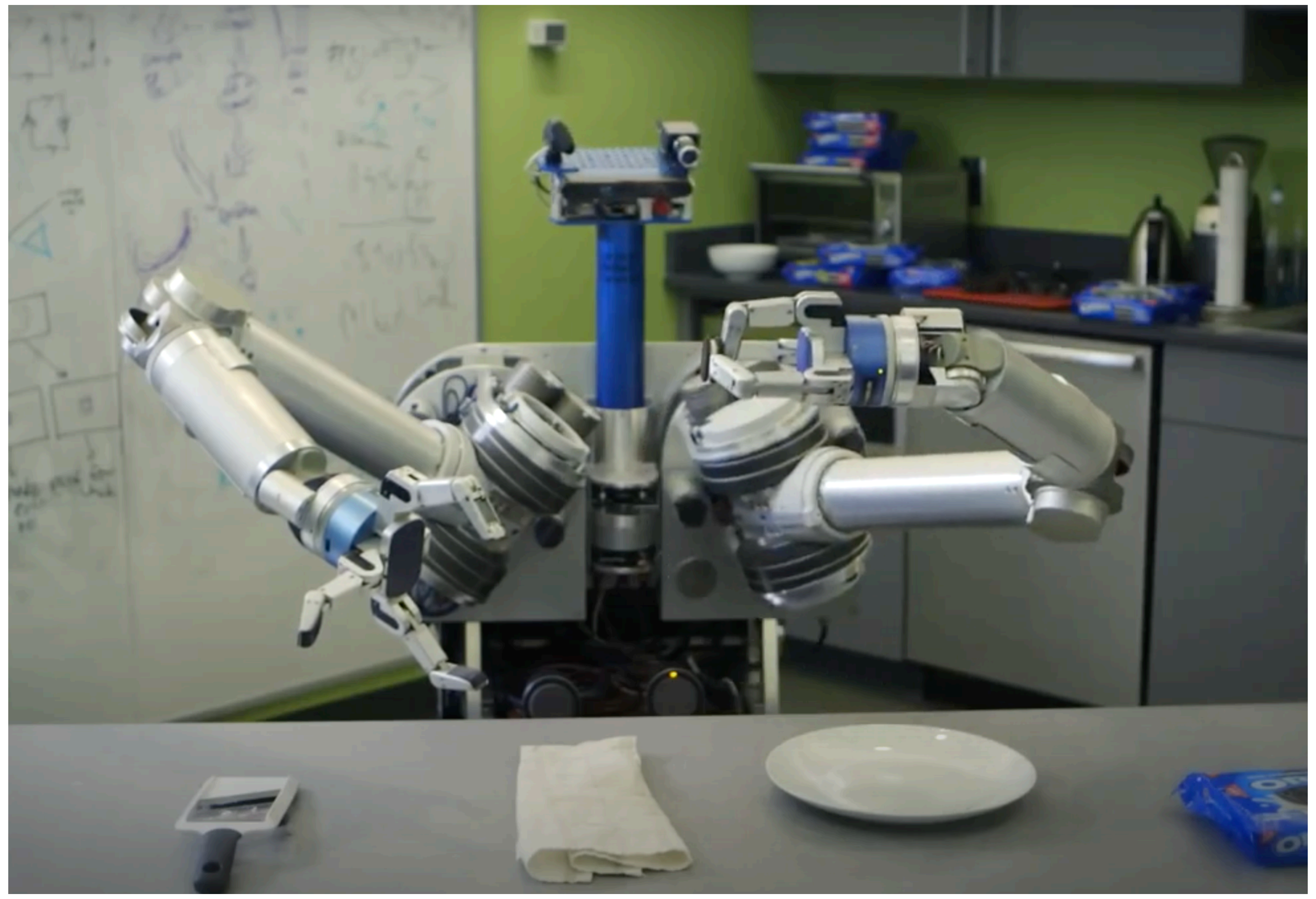
Optimal rate for **reward-aware** RL = $\Theta\left(\frac{SA}{\epsilon^2}\right)$

Optimal rate for **reward-free** RL = $\Theta\left(\frac{S^2A}{\epsilon^2}\right)$



Motivation

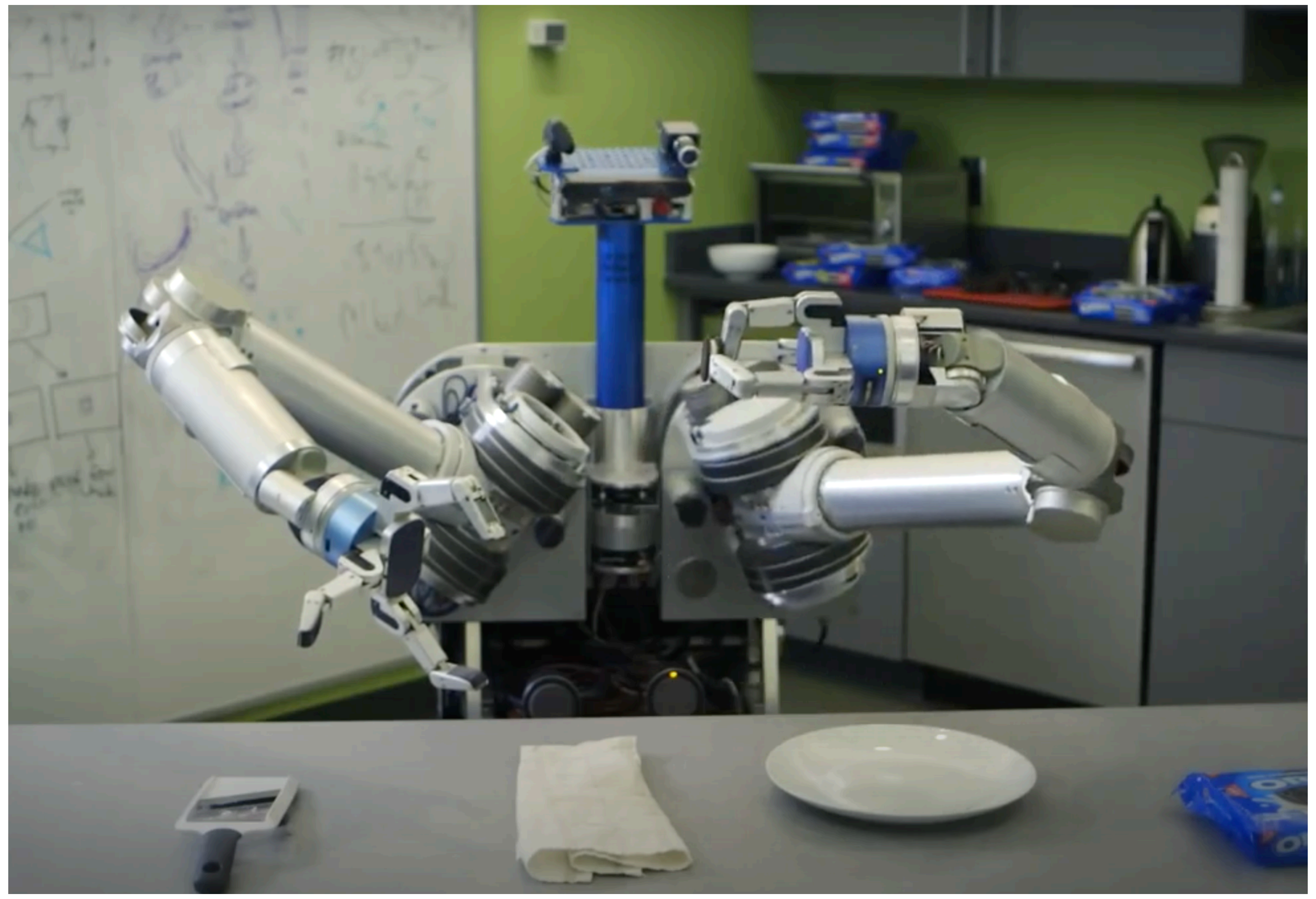
In real-world settings, state spaces are often large or infinite, and we must turn to **function approximation**



Motivation

In real-world settings, state spaces are often large or infinite, and we must turn to **function approximation**

Is reward-free RL harder than reward-aware RL in MDPs with large state-spaces?



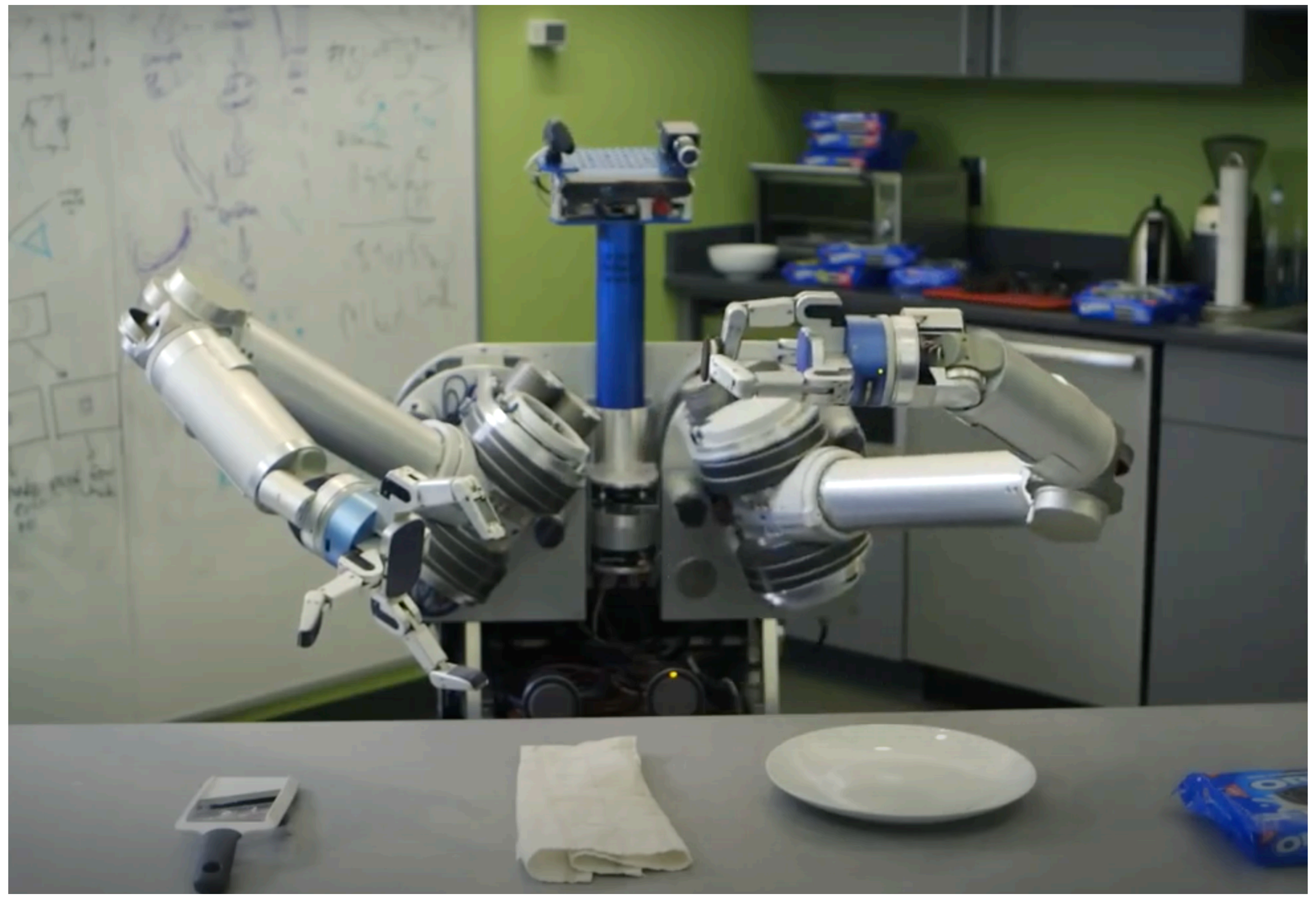
Motivation

In real-world settings, state spaces are often large or infinite, and we must turn to **function approximation**

Is reward-free RL harder than reward-aware RL in MDPs with large state-spaces?

We consider **linear MDPs** (Jin et al., 2020), parameterized by d-dimensional feature vectors ϕ:

$$P_h(s' | s, a) = \langle \phi(s, a), \mu_h(s') \rangle, \quad r_h(s, a) = \langle \phi(s, a), \theta_h \rangle$$



Our Contributions

In the setting of Linear MDPs, we develop a computationally efficient **reward-free** algorithm with complexity of:

$$\tilde{\mathcal{O}}\left(d^2 H^5/\epsilon^2\right)$$

Our Contributions

In the setting of Linear MDPs, we develop a computationally efficient **reward-free** algorithm with complexity of:

$$\tilde{\mathcal{O}}\left(d^2H^5/\epsilon^2\right)$$

We show a lower bound on **reward-aware RL** of:

$$\Omega\left(d^2H^2/\epsilon^2\right)$$

Our Contributions

In the setting of Linear MDPs, we develop a computationally efficient **reward-free** algorithm with complexity of:

$$\tilde{\mathcal{O}}\left(d^2H^5/\epsilon^2\right)$$

We show a lower bound on **reward-aware RL** of:

$$\Omega\left(d^2H^2/\epsilon^2\right)$$

Our results imply the surprising conclusion that, up to H factors, **reward-free RL is no harder than reward-aware RL in linear MDPs**

Our Contributions

In the setting of Linear MDPs, we develop a computationally efficient **reward-free** algorithm with complexity of:

$$\tilde{\mathcal{O}}\left(d^2H^5/\epsilon^2\right)$$

We show a lower bound on **reward-aware RL** of:

$$\Omega\left(d^2H^2/\epsilon^2\right)$$

Our results imply the surprising conclusion that, up to H factors, **reward-free RL is no harder than reward-aware RL in linear MDPs**

Our results are the first **dimension-optimal, computationally efficient** bounds for linear MDPs

Algorithm

Algorithm

Phase 1 (Exploration)

Algorithm

Phase 1 (Exploration)

Assume we have collected covariates $\Lambda_k = \sum_{\tau=1}^{K} \phi_\tau \phi_\tau^\top + \lambda I$, set:

$$r^k(\phi) \sim \|\phi\|^2_{\Lambda_k^{-1}}$$

Algorithm

Phase 1 (Exploration)

Assume we have collected covariates $\Lambda_k = \sum_{\tau=1}^{K} \phi_\tau \phi_\tau^\top + \lambda I$, set:

$$r^k(\phi) \sim \|\phi\|^2_{\Lambda_k^{-1}}$$

Run regret minimization algorithm on r^k to incentivize exploration

Algorithm

Phase 1 (Exploration)

Assume we have collected covariates $\Lambda_k = \sum_{\tau=1}^{K} \phi_\tau \phi_\tau^\top + \lambda I$, set:

$$r^k(\phi) \sim \|\phi\|^2_{\Lambda_k^{-1}}$$

Run regret minimization algorithm on r^k to incentivize exploration

Key Idea: If we run a *first-order* regret minimization algorithm, the cost of "learning to explore" is absorbed in a lower-order $\mathcal{O}(1/\epsilon)$ term

Algorithm

Phase 1 (Exploration)

Assume we have collected covariates $\Lambda_k = \sum_{\tau=1}^{K} \phi_\tau \phi_\tau^\top + \lambda I$, set:

$$r^k(\phi) \sim \|\phi\|^2_{\Lambda_k^{-1}}$$

Run regret minimization algorithm on r^k to incentivize exploration

Key Idea: If we run a *first-order* regret minimization algorithm, the cost of "learning to explore" is absorbed in a lower-order $\mathcal{O}(1/\epsilon)$ term

Phase 2 (Policy Construction)

Algorithm

Phase 1 (Exploration)

Assume we have collected covariates $\Lambda_k = \sum_{\tau=1}^{K} \phi_\tau \phi_\tau^\top + \lambda I$, set:

$$r^k(\phi) \sim \|\phi\|^2_{\Lambda_k^{-1}}$$

Run regret minimization algorithm on r^k to incentivize exploration

Key Idea: If we run a *first-order* regret minimization algorithm, the cost of "learning to explore" is absorbed in a lower-order $\mathcal{O}(1/\epsilon)$ term

Phase 2 (Policy Construction)

Given data from exploring, construct an "optimistic" policy using a least-squares value-iteration procedure

Thanks!