

Bellman Rank

Kianté Brantley & Sham Kakade

Outline

- 1 The Exploration Problem
- 2 A Natural Algorithm: Optimistic Elimination
- 3 The Structural Bottleneck: Bellman Rank
- 4 The Model Zoo
- 5 The Sample-Based Reality

Recap: The "Bonus" Recipe for Exploration

So far, we have a clear recipe for strategic exploration: **build an explicit uncertainty bonus.**

- **Tabular UCBVI:** Bonus $\propto 1/\sqrt{N(s, a)}$. Works, but scales with $|\mathcal{S}||\mathcal{A}|$.
- **Linear MDPs:** Bonus $\propto \|\phi(s, a)\|_{\Lambda^{-1}}$. Scales with d .

Recap: The "Bonus" Recipe for Exploration

So far, we have a clear recipe for strategic exploration: **build an explicit uncertainty bonus.**

- **Tabular UCBVI:** Bonus $\propto 1/\sqrt{N(s, a)}$. Works, but scales with $|\mathcal{S}||\mathcal{A}|$.
- **Linear MDPs:** Bonus $\propto \|\phi(s, a)\|_{\Lambda^{-1}}$. Scales with d .

The Catch: Both rely on explicitly estimating the transition dynamics (either locally or via known features) to construct the bonus.

Recap: The "Bonus" Recipe for Exploration

So far, we have a clear recipe for strategic exploration: **build an explicit uncertainty bonus.**

- **Tabular UCBVI:** Bonus $\propto 1/\sqrt{N(s, a)}$. Works, but scales with $|\mathcal{S}||\mathcal{A}|$.
- **Linear MDPs:** Bonus $\propto \|\phi(s, a)\|_{\Lambda^{-1}}$. Scales with d .

The Catch: Both rely on explicitly estimating the transition dynamics (either locally or via known features) to construct the bonus.

What if we only have Linear Bellman Completeness?

$$r_h(s, a) + \mathbb{E}_{P^*}[\max_{a'} w_{h+1} \cdot \phi(s', a')] = \phi(s, a)^\top \mathcal{T}_h(w_{h+1})$$

$\in \mathbb{R}^d$

We know this is learnable with a *generative model* (simulator). But how do we explore *online*?

Our Goal: PAC-RL for General Classes

We want to move beyond explicit bonuses and ask: when can we explore efficiently given an arbitrary, structured hypothesis class \mathcal{H} ?

Our Goal: PAC-RL for General Classes

We want to move beyond explicit bonuses and ask: when can we explore efficiently given an arbitrary, structured hypothesis class \mathcal{H} ?

The PAC-RL Objective (Fixed Confidence): Given $\epsilon, \delta \in (0, 1)$, we want to output a *single* policy $\hat{\pi}$ such that:

$$V_0^*(s_0) - V_0^{\hat{\pi}}(s_0) \leq \epsilon \quad \text{with probability at least } 1 - \delta.$$

Our Goal: PAC-RL for General Classes

We want to move beyond explicit bonuses and ask: when can we explore efficiently given an arbitrary, structured hypothesis class \mathcal{H} ?

The PAC-RL Objective (Fixed Confidence): Given $\epsilon, \delta \in (0, 1)$, we want to output a *single* policy $\hat{\pi}$ such that:

$$V_0^*(s_0) - V_0^{\hat{\pi}}(s_0) \leq \epsilon \quad \text{with probability at least } 1 - \delta.$$

PAC vs. Regret:

- **Regret:** Control cumulative suboptimality *during* learning. Requires carefully shrinking uncertainty bonuses at every step.
- **PAC:** We just need to find one good policy eventually. This allows for purely *elimination-based* strategies.

Outline

- 1 The Exploration Problem
- 2 A Natural Algorithm: Optimistic Elimination**
- 3 The Structural Bottleneck: Bellman Rank
- 4 The Model Zoo
- 5 The Sample-Based Reality

The Core Object: Average Bellman Error

Data reuse: We can't build a structural bonus, so let's test if our hypotheses are consistent with the collected data.

Define the one-step **Bellman residual** of a hypothesis $g \in \mathcal{H}$ as:

$$\mathcal{H} = \{ f(s, a) \}$$

$$\ell_h(s, a, s'; g) := Q_{h,g}(s, a) - r_h(s, a) - V_{h+1,g}(s')$$

$$\stackrel{\text{max}}{g'} Q_{h+1}(s', a')$$

The Core Object: Average Bellman Error

Data reuse: We can't build a structural bonus, so let's test if our hypotheses are consistent with the collected data.

Define the one-step **Bellman residual** of a hypothesis $g \in \mathcal{H}$ as:

$$\mathbb{E}_{s' \sim P_{s'}} \left[\ell_h(s, a, s'; g) \right] := \mathbb{E} \left[Q_{h,g}(s, a) - r_h(s, a) - V_{h+1,g}(s') \right]$$

With data from some “roll in policy” π_f , we can check if the average Bellman error is small.

We evaluate candidate g using its **Average Bellman Error** under the roll-in distribution induced by another policy π_f :

$$\mathcal{E}_h(f, g) := \mathbb{E}_{(s,a) \sim d_h^{\pi_f}, s' \sim P_h^*} \left[\ell_h(s, a, s'; g) \right]$$

The Core Object: Average Bellman Error

Data reuse: We can't build a structural bonus, so let's test if our hypotheses are consistent with the collected data.

Define the one-step **Bellman residual** of a hypothesis $g \in \mathcal{H}$ as:

$$\ell_h(s, a, s'; g) := Q_{h,g}(s, a) - r_h(s, a) - V_{h+1,g}(s')$$

With data from some “roll in policy” π_f , we can check if the average Bellman error is small.

We evaluate candidate g using its **Average Bellman Error** under the roll-in distribution induced by another policy π_f :

$$\mathcal{E}_h(f, g) := \mathbb{E}_{(s,a) \sim d_h^{\pi_f}, s' \sim P_h^*} [\ell_h(s, a, s'; g)]$$

Key Insight: Under realizability, the true optimal hypothesis $f^ \in \mathcal{H}$ is perfectly consistent everywhere. Thus, $\mathcal{E}_h(f, f^*) = 0$ for every roll-in policy f .*

Data Reuse: Estimating Errors for all $g \in \mathcal{H}$

How do we actually estimate $\mathcal{E}_h(f, g)$ from data?

1. Data Collection: Fix a roll-in hypothesis f . Execute its greedy policy π_f for m independent episodes. At stage h , this yields a dataset:

$$\mathcal{D}_{f,h} = \left\{ (s_h^{(i)}, a_h^{(i)}, s_{h+1}^{(i)}) \right\}_{i=1}^m$$

Data Reuse: Estimating Errors for all $g \in \mathcal{H}$

How do we actually estimate $\mathcal{E}_h(f, g)$ from data?

1. Data Collection: Fix a roll-in hypothesis f . Execute its greedy policy π_f for m independent episodes. At stage h , this yields a dataset:

$$\mathcal{D}_{f,h} = \left\{ (s_h^{(i)}, a_h^{(i)}, s_{h+1}^{(i)}) \right\}_{i=1}^m$$

2. The Empirical Estimator: For *any* candidate $g \in \mathcal{H}$, we define its empirical Bellman error on this dataset as:

$$\hat{\mathcal{E}}_h(f, g) := \frac{1}{m} \sum_{i=1}^m \left(Q_{h,g}(s_h^{(i)}, a_h^{(i)}) - r_h(s_h^{(i)}, a_h^{(i)}) - V_{h+1,g}(s_{h+1}^{(i)}) \right)$$

Data Reuse: Estimating Errors for all $g \in \mathcal{H}$

How do we actually estimate $\mathcal{E}_h(f, g)$ from data?

1. Data Collection: Fix a roll-in hypothesis f . Execute its greedy policy π_f for m independent episodes. At stage h , this yields a dataset:

$$\mathcal{D}_{f,h} = \left\{ (s_h^{(i)}, a_h^{(i)}, s_{h+1}^{(i)}) \right\}_{i=1}^m$$

estimate

$\mathcal{E}_h(f, g)$ by

2. The Empirical Estimator: For *any* candidate $g \in \mathcal{H}$, we define its empirical Bellman error on this dataset as:

$$\hat{\mathcal{E}}_h(f, g) := \frac{1}{m} \sum_{i=1}^m \left(Q_{h,g}(s_h^{(i)}, a_h^{(i)}) - r_h(s_h^{(i)}, a_h^{(i)}) - V_{h+1,g}(s_{h+1}^{(i)}) \right)$$

The Supervised Learning (SL) Connection: Notice that the dataset $\mathcal{D}_{f,h}$ depends *only* on the roll-in policy f . It does not depend on g .

- We collect data **once** using f .
- We can then evaluate $\hat{\mathcal{E}}_h(f, g)$ for **every single** $g \in \mathcal{H}$ offline.

Algorithm: Optimistic Elimination (Sample-Based)

Algorithm Loop: For iteration $t = 1, 2, \dots$

- 1 **Version Space:** Keep all hypotheses $g \in \mathcal{H}$ that have small empirical error on all past datasets:

$$|\widehat{\mathcal{E}}_h(f_i, g)| \leq \epsilon_{\text{stat}} \quad \forall i < t, \quad \forall h$$

- 2 **Optimism:** Pick the surviving hypothesis f_t that predicts the highest value $V_{0,g}(s_0)$.
- 3 **Roll-in:** Gather m trajectories using π_{f_t} to form dataset \mathcal{D}_t .

Algorithm: Optimistic Elimination (Sample-Based)

$$\{ g: \hat{\mathcal{E}}_n(f_1, g) \approx 0, \hat{\mathcal{E}}_n(f_2, g) \approx 0, \dots, \hat{\mathcal{E}}_n(f_{t-1}, g) \approx 0 \}$$

Algorithm Loop: For iteration $t = 1, 2, \dots$

- 1 **Version Space:** Keep all hypotheses $g \in \mathcal{H}$ that have small empirical error on all past datasets:

$$|\hat{\mathcal{E}}_h(f_i, g)| \leq \epsilon_{\text{stat}} \quad \forall i < t, \quad \forall h$$

- 2 **Optimism:** Pick the surviving hypothesis f_t that predicts the highest value $V_{0,g}(s_0)$.
- 3 **Roll-in:** Gather m trajectories using π_{f_t} to form dataset \mathcal{D}_t .

The Power of Data Reuse: We use \mathcal{D}_t (collected via f_t) to evaluate the empirical average $\hat{\mathcal{E}}_h(f_t, g)$ for all $g \in \mathcal{H}$. This reduces RL exploration to a sequence of Supervised Learning-style risk minimization problems.

Why does this work? (Lemma 1: Self-Error)

To understand why this works, we need to connect a hypothesis's *error* to its actual *performance*.

Lemma (Self-Error Telescoping Identity)

For any $f \in \mathcal{H}$, the difference between its predicted value and the true value of its greedy policy π_f is exactly the sum of its self-errors:

$$\max_{a_0} f(s_0, a_0) - V_0^{\pi_f}(s_0) = V_{0,f}(s_0) - V_0^{\pi_f}(s_0) = \sum_{h=0}^{H-1} \mathcal{E}_h(f, f)$$

Why does this work? (Lemma 1: Self-Error)

To understand why this works, we need to connect a hypothesis's *error* to its actual *performance*.

Lemma (Self-Error Telescoping Identity)

For any $f \in \mathcal{H}$, the difference between its predicted value and the true value of its greedy policy π_f is exactly the sum of its self-errors:

$$V_{0,f}(s_0) - V_0^{\pi_f}(s_0) = \sum_{h=0}^{H-1} \mathcal{E}_h(f, f)$$

Note: $V_{0,f}(s_0) = \max_a Q_{0,f}(s_0, a)$ is what the hypothesis f **claims** it will get. $V_0^{\pi_f}(s_0)$ is what it **actually** gets in the real environment.

Proof of Lemma 1

Proof: Take the expectation over the true trajectory (s_h, a_h) induced by running π_f . Since the policy takes the greedy action, $a_h = \pi_{h,f}(s_h)$, we have $V_{h,f}(s_h) = Q_{h,f}(s_h, a_h)$.

Proof of Lemma 1

Proof: Take the expectation over the true trajectory (s_h, a_h) induced by running π_f . Since the policy takes the greedy action, $a_h = \pi_{h,f}(s_h)$, we have $V_{h,f}(s_h) = Q_{h,f}(s_h, a_h)$.

$$V_{H+1}(s) = 0$$

$$V_{0,f}(s_0) - \sum_{h=0}^{H-1} r_h(s_h, a_h) = \sum_{h=0}^{H-1} \left(V_{h,f}(s_h) - r_h(s_h, a_h) - V_{h+1,f}(s_{h+1}) \right)$$

Proof of Lemma 1

Proof: Take the expectation over the true trajectory (s_h, a_h) induced by running π_f . Since the policy takes the greedy action, $a_h = \pi_{h,f}(s_h)$, we have $V_{h,f}(s_h) = Q_{h,f}(s_h, a_h)$.

$$\begin{aligned} V_{0,f}(s_0) - \sum_{h=0}^{H-1} r_h(s_h, a_h) &= \sum_{h=0}^{H-1} \left(V_{h,f}(s_h) - r_h(s_h, a_h) - V_{h+1,f}(s_{h+1}) \right) \\ &= \sum_{h=0}^{H-1} \left(Q_{h,f}(s_h, a_h) - r_h(s_h, a_h) - V_{h+1,f}(s_{h+1}) \right) \end{aligned}$$

Proof of Lemma 1

Proof: Take the expectation over the true trajectory (s_h, a_h) induced by running π_f . Since the policy takes the greedy action, $a_h = \pi_{h,f}(s_h)$, we have $V_{h,f}(s_h) = Q_{h,f}(s_h, a_h)$.

$$\begin{aligned} V_{0,f}(s_0) - \sum_{h=0}^{H-1} r_h(s_h, a_h) &= \sum_{h=0}^{H-1} \left(V_{h,f}(s_h) - r_h(s_h, a_h) - V_{h+1,f}(s_{h+1}) \right) \\ &= \sum_{h=0}^{H-1} \left(Q_{h,f}(s_h, a_h) - r_h(s_h, a_h) - V_{h+1,f}(s_{h+1}) \right) \end{aligned}$$

Taking the expectation over the trajectory dynamics yields exactly the sum of the average Bellman errors: $\sum_{h=0}^{H-1} \mathcal{E}_h(f, f)$. □

OE Correctness: The Exact Case

Assume the exact case for a moment ($\epsilon_{\text{stat}} = 0$). Does this algorithm find the optimal policy?

Assume $f^* \in \mathcal{H}$.

- 1 **Survival:** f^* is never eliminated because $\mathcal{E}_h(f, f^*) = 0$ for all roll-ins f .

OE Correctness: The Exact Case

Assume the exact case for a moment ($\epsilon_{\text{stat}} = 0$). Does this algorithm find the optimal policy?

Assume $f^* \in \mathcal{H}$.

$$J^* = Q^*$$

- 1 **Survival:** f^* is never eliminated because $\mathcal{E}_h(f, f^*) = 0$ for all roll-ins f .
- 2 **Optimism:** Because f^* is in the version space, the algorithm picks an f_t such that $V_{0, f_t}(s_0) \geq V_{0, f^*}(s_0) = V^*(s_0)$.

$$\Rightarrow V_{0, f_t} - V^{\pi_{f_t}} \neq 0$$

OE Correctness: The Exact Case

Assume the exact case for a moment ($\epsilon_{\text{stat}} = 0$). Does this algorithm find the optimal policy?

Assume $f^* \in \mathcal{H}$.

- 1 **Survival:** f^* is never eliminated because $\mathcal{E}_h(f, f^*) = 0$ for all roll-ins f .
- 2 **Optimism:** Because f^* is in the version space, the algorithm picks an f_t such that $V_{0, f_t}(s_0) \geq V_{0, f^*}(s_0) = V^*(s_0)$.
- 3 **Elimination:** Suppose f_t is strictly sub-optimal ($V^*(s_0) > V^{\pi_{f_t}}(s_0)$).
 - By optimism, $V_{0, f_t}(s_0) - V^{\pi_{f_t}}(s_0) > 0$.
 - By Lemma 1, this means $\sum_h \mathcal{E}_h(f_t, f_t) > 0$.
 - Therefore, f_t will have a non-zero error on its own dataset \mathcal{D}_t , and will be **eliminated** in the next round.

OE Correctness: The Exact Case

Assume the exact case for a moment ($\epsilon_{\text{stat}} = 0$). Does this algorithm find the optimal policy?

Assume $f^* \in \mathcal{H}$.

- ① **Survival:** f^* is never eliminated because $\mathcal{E}_h(f, f^*) = 0$ for all roll-ins f .
- ② **Optimism:** Because f^* is in the version space, the algorithm picks an f_t such that $V_{0, f_t}(s_0) \geq V_{0, f^*}(s_0) = V^*(s_0)$.
- ③ **Elimination:** Suppose f_t is strictly sub-optimal ($V^*(s_0) > V^{\pi_{f_t}}(s_0)$).
 - By optimism, $V_{0, f_t}(s_0) - V^{\pi_{f_t}}(s_0) > 0$.
 - By Lemma 1, this means $\sum_h \mathcal{E}_h(f_t, f_t) > 0$.
 - Therefore, f_t will have a non-zero error on its own dataset \mathcal{D}_t , and will be **eliminated** in the next round.

Conclusion: Every round we either play the optimal policy, or we eliminate at least one hypothesis. The algorithm terminates in trivially $\leq |\mathcal{H}|$ rounds.

The Big Question

We established a clean algorithmic loop:

- Pick an optimistic hypothesis f_t .
- Roll it out to collect data.
- If it is sub-optimal, it generates a non-zero self-error $\mathcal{E}_h(f_t, f_t) \neq 0$.
- We add this data to our test set, ensuring f_t is eliminated forever.

The Big Question

We established a clean algorithmic loop:

- Pick an optimistic hypothesis f_t .
- Roll it out to collect data.
- If it is sub-optimal, it generates a non-zero self-error $\mathcal{E}_h(f_t, f_t) \neq 0$.
- We add this data to our test set, ensuring f_t is eliminated forever.

Jacob L. Rosen
 $\int \pi | = A^S$

The Catch: We proved it takes at most $|\mathcal{H}|$ rounds. But if \mathcal{H} is massive, or a continuous space of neural networks, $|\mathcal{H}|$ could be infinite.

The Big Question

We established a clean algorithmic loop:

- Pick an optimistic hypothesis f_t .
- Roll it out to collect data.
- If it is sub-optimal, it generates a non-zero self-error $\mathcal{E}_h(f_t, f_t) \neq 0$.
- We add this data to our test set, ensuring f_t is eliminated forever.

The Catch: We proved it takes at most $|\mathcal{H}|$ rounds. But if \mathcal{H} is massive, or a continuous space of neural networks, $|\mathcal{H}|$ could be infinite.

*To guarantee this terminates quickly independent of $|\mathcal{H}|$, we want a structural assumption about the **matrix of errors**... the **Bellman Rank**.*

Outline

- 1 The Exploration Problem
- 2 A Natural Algorithm: Optimistic Elimination
- 3 The Structural Bottleneck: Bellman Rank**
- 4 The Model Zoo
- 5 The Sample-Based Reality

The Error Matrix

Consider a matrix M_h where the rows correspond to roll-in policies $f \in \mathcal{H}$ and the columns correspond to tested hypotheses $g \in \mathcal{H}$.

Each entry is the cross-error:

$$M_h[f, g] := \mathcal{E}_h(f, g)$$

The Error Matrix

Consider a matrix M_h where the rows correspond to roll-in policies $f \in \mathcal{H}$ and the columns correspond to tested hypotheses $g \in \mathcal{H}$.

Each entry is the cross-error:

$$M_h[f, g] := \mathcal{E}_h(f, g)$$

If \mathcal{H} is large or continuous, M_h is massive. However, in many reinforcement learning settings, this matrix contains redundant information.

We formalize this redundancy by assuming M_h has a low rank.

Defining Bellman Rank

Definition (Q -Bellman Rank)

The environment and hypothesis class pair $(\mathcal{M}, \mathcal{H})$ has Q -Bellman rank at most d if for each stage h there exist maps $X_h, W_h : \mathcal{H} \rightarrow \mathbb{R}^d$ such that:

$$\mathcal{E}_h(f, g) = \langle X_h(f), W_h(g) \rangle \quad \forall f, g \in \mathcal{H}$$

Defining Bellman Rank

Definition (Q-Bellman Rank)

The environment and hypothesis class pair $(\mathcal{M}, \mathcal{H})$ has *Q-Bellman rank at most d* if for each stage h there exist maps $X_h, W_h : \mathcal{H} \rightarrow \mathbb{R}^d$ such that:

$$\mathcal{E}_h(f, g) = \langle X_h(f), W_h(g) \rangle \quad \forall f, g \in \mathcal{H}$$

Key Properties:

- $X_h(f)$ acts as the latent feature of the roll-in distribution $d_h^{\pi_f}$.
- $W_h(g)$ acts as the latent feature of the Bellman residual of g .
- We only assume these maps exist. The algorithm does not need to know X_h or W_h to execute Optimistic Elimination.

$$\begin{aligned} \mathbb{E}_{x \sim p} [l(x)] \\ = \vec{v} \cdot \vec{l} \end{aligned}$$

Fast Convergence via Dimension Counting

Let us return to the exact Optimistic Elimination algorithm ($\epsilon_{\text{stat}} = 0$). How many rounds does it take to terminate if the Bellman Rank is d ?

Fast Convergence via Dimension Counting

Let us return to the exact Optimistic Elimination algorithm ($\epsilon_{\text{stat}} = 0$). How many rounds does it take to terminate if the Bellman Rank is d ?

Suppose at iteration t , the algorithm selects f_t , and it is strictly sub-optimal.

- 1 By the Self-Error Telescoping Identity (Lemma 1), a sub-optimal optimistic hypothesis must have a non-zero self-error at some stage h :

$$\mathcal{E}_h(f_t, f_t) \neq 0$$

Fast Convergence via Dimension Counting

Let us return to the exact Optimistic Elimination algorithm ($\epsilon_{\text{stat}} = 0$). How many rounds does it take to terminate if the Bellman Rank is d ?

Suppose at iteration t , the algorithm selects f_t , and it is strictly sub-optimal.

- 1 By the Self-Error Telescoping Identity (Lemma 1), a sub-optimal optimistic hypothesis must have a non-zero self-error at some stage h :

$$\mathcal{E}_h(f_t, f_t) \neq 0$$

- 2 Using the Bellman rank factorization, this implies:

$$\langle X_h(f_t), W_h(f_t) \rangle \neq 0$$

The Geometry of Elimination

- ③ Because f_t survived the version space, it must have had zero error on all data collected in previous rounds $i < t$:

$$\mathcal{E}_h(f_i, f_t) = 0 \implies \langle X_h(f_i), W_h(f_t) \rangle = 0 \quad \forall i < t$$

The Geometry of Elimination

- ③ Because f_t survived the version space, it must have had zero error on all data collected in previous rounds $i < t$:

$$\mathcal{E}_h(f_i, f_t) = 0 \implies \langle X_h(f_i), W_h(f_t) \rangle = 0 \quad \forall i < t$$

- ④ This means $W_h(f_t)$ is orthogonal to the subspace spanned by all previous roll-in features $\{X_h(f_1), \dots, X_h(f_{t-1})\}$.

The Geometry of Elimination

- 3 Because f_t survived the version space, it must have had zero error on all data collected in previous rounds $i < t$:

$$\mathcal{E}_h(f_i, f_t) = 0 \implies \langle X_h(f_i), W_h(f_t) \rangle = 0 \quad \forall i < t$$

- 4 This means $W_h(f_t)$ is orthogonal to the subspace spanned by all previous roll-in features $\{X_h(f_1), \dots, X_h(f_{t-1})\}$.
- 5 Since $\langle X_h(f_t), W_h(f_t) \rangle \neq 0$, the new roll-in feature $X_h(f_t)$ cannot be entirely contained in that past subspace. It must possess a linearly independent component.

The Geometry of Elimination

- 3 Because f_t survived the version space, it must have had zero error on all data collected in previous rounds $i < t$:

$$\mathcal{E}_h(f_i, f_t) = 0 \implies \langle X_h(f_i), W_h(f_t) \rangle = 0 \quad \forall i < t$$

- 4 This means $W_h(f_t)$ is orthogonal to the subspace spanned by all previous roll-in features $\{X_h(f_1), \dots, X_h(f_{t-1})\}$.
- 5 Since $\langle X_h(f_t), W_h(f_t) \rangle \neq 0$, the new roll-in feature $X_h(f_t)$ cannot be entirely contained in that past subspace. It must possess a linearly independent component.

Conclusion: Every time we execute a sub-optimal policy, we discover a strictly new, linearly independent direction in \mathbb{R}^d . In a d -dimensional space, this can happen at most d times per stage. The algorithm terminates in at most dH rounds.

Outline

- 1 The Exploration Problem
- 2 A Natural Algorithm: Optimistic Elimination
- 3 The Structural Bottleneck: Bellman Rank
- 4 The Model Zoo**
- 5 The Sample-Based Reality

Example 1: Linear Bellman Completeness

- **Hypothesis Class \mathcal{H} :** $Q_{h,w}(s, a) = \phi(s, a)^\top w_h$, where $\|w_h\|_2 \leq W$.
- **Completeness Assumption:** $r_h(s, a) + \mathbb{E}_{P^*}[\max_{a'} w_{h+1}^\top \phi(s', a')] = \phi(s, a)^\top \mathcal{T}_h(w_{h+1})$.

Example 1: Linear Bellman Completeness

- **Hypothesis Class \mathcal{H} :** $Q_{h,w}(s, a) = \phi(s, a)^\top w_h$, where $\|w_h\|_2 \leq W$.
- **Completeness Assumption:** $r_h(s, a) + \mathbb{E}_{P^*}[\max_{a'} w_{h+1}^\top \phi(s', a')] = \phi(s, a)^\top \mathcal{T}_h(w_{h+1})$.

The Factorization: $f = \tilde{w}$, $g = w$, $\pi_{\tilde{w}}(s) = \operatorname{argmax}_a \tilde{w}_h^\top \phi(s, a)$.

$$\begin{aligned} \mathcal{E}_h(\tilde{w}, w) &= \mathbb{E}_{d_h^{\pi_{\tilde{w}}}} \left[\phi(s, a)^\top w_h - \phi(s, a)^\top \mathcal{T}_h(w_{h+1}) \right] \\ &= \left\langle \underbrace{\mathbb{E}_{d_h^{\pi_{\tilde{w}}}}[\phi(s, a)]}_{X_h(\tilde{w})}, \underbrace{w_h - \mathcal{T}_h(w_{h+1})}_{W_h(w)} \right\rangle \end{aligned}$$

Example 1: Linear Bellman Completeness

- **Hypothesis Class \mathcal{H} :** $Q_{h,w}(s, a) = \phi(s, a)^\top w_h$, where $\|w_h\|_2 \leq W$.
- **Completeness Assumption:** $r_h(s, a) + \mathbb{E}_{P^*}[\max_{a'} w_{h+1}^\top \phi(s', a')] = \phi(s, a)^\top \mathcal{T}_h(w_{h+1})$.

The Factorization: $f = \tilde{w}$, $g = w$, $\pi_{\tilde{w}}(s) = \operatorname{argmax}_a \tilde{w}_h^\top \phi(s, a)$.

$$\begin{aligned} \mathcal{E}_h(\tilde{w}, w) &= \mathbb{E}_{d_h^{\pi_{\tilde{w}}}} \left[\phi(s, a)^\top w_h - \phi(s, a)^\top \mathcal{T}_h(w_{h+1}) \right] \\ &= \left\langle \underbrace{\mathbb{E}_{d_h^{\pi_{\tilde{w}}}}[\phi(s, a)]}_{X_h(\tilde{w})}, \underbrace{w_h - \mathcal{T}_h(w_{h+1})}_{W_h(w)} \right\rangle \end{aligned}$$

Key Observations:

- This implies a Bellman Rank of at most d .
- This uses the exact same feature ϕ as the Linear MDP analysis.
- **Crucially:** Optimistic Elimination only requires access to \mathcal{H} , and not $X(\cdot)$.

Example 2: Sparse Linear MDPs

Suppose the features are extremely high-dimensional: $\phi(s, a) \in \mathbb{R}^D$, where $D \gg |\mathcal{S}|$.

Assume the environment is a Linear MDP, but the transition dynamics depend entirely on an *unknown* active subset of s features ($s \ll D$).

Example 2: Sparse Linear MDPs

Suppose the features are extremely high-dimensional: $\phi(s, a) \in \mathbb{R}^D$, where $D \gg |\mathcal{S}|$.

Assume the environment is a Linear MDP, but the transition dynamics depend entirely on an *unknown* active subset of s features ($s \ll D$).

Hypothesis Class \mathcal{H} :

- We restrict our search to linear models using at most s features.
- $\mathcal{H} = \bigcup_{K \subset [D], |K|=s} \mathcal{H}(\phi_K)$, where $\mathcal{H}(\phi_K)$ is the dense class restricted to the subset K .

Example 2: Sparse Linear MDPs

Suppose the features are extremely high-dimensional: $\phi(s, a) \in \mathbb{R}^D$, where $D \gg |\mathcal{S}|$.

Assume the environment is a Linear MDP, but the transition dynamics depend entirely on an *unknown* active subset of s features ($s \ll D$).

Hypothesis Class \mathcal{H} :

- We restrict our search to linear models using at most s features.
- $\mathcal{H} = \bigcup_{K \subset [D], |K|=s} \mathcal{H}(\phi_K)$, where $\mathcal{H}(\phi_K)$ is the dense class restricted to the subset K .

What is the Bellman Rank?

- The ambient dimension is D .
- However, the transition operator factors through an s -dimensional bottleneck.
- Therefore, the latent Bellman Rank is s .

Note, we do not need to know the active subset here.

Example 2: Sparse Linear MDPs

Suppose the features are extremely high-dimensional: $\phi(s, a) \in \mathbb{R}^D$, where $D \gg |\mathcal{S}|$.

Assume the environment is a Linear MDP, but the transition dynamics depend entirely on an *unknown* active subset of s features ($s \ll D$).

Hypothesis Class \mathcal{H} :

- We restrict our search to linear models using at most s features.
- $\mathcal{H} = \bigcup_{K \subset [D], |K|=s} \mathcal{H}(\phi_K)$, where $\mathcal{H}(\phi_K)$ is the dense class restricted to the subset K .

What is the Bellman Rank?

- The ambient dimension is D .
- However, the transition operator factors through an s -dimensional bottleneck.
- Therefore, the latent Bellman Rank is s .

Note, we do not need to know the active subset here.

Note: Applying Optimistic Elimination here involves some subtleties regarding off-policy evaluation (V-Bellman Rank), but the core geometric structure remains a rank- s error matrix.

Outline

- 1 The Exploration Problem
- 2 A Natural Algorithm: Optimistic Elimination
- 3 The Structural Bottleneck: Bellman Rank
- 4 The Model Zoo
- 5 The Sample-Based Reality

From Exact to Sample-Based Elimination

We estimate $\hat{\mathcal{E}}_h(f_i, g)$ using finite datasets of size m .

Assumption (Realizability): The true optimal hypothesis $f^* \in \mathcal{H}$.

From Exact to Sample-Based Elimination

We estimate $\widehat{\mathcal{E}}_h(f_i, g)$ using finite datasets of size m .

Assumption (Realizability): The true optimal hypothesis $f^* \in \mathcal{H}$.

The Statistical Relaxation: Because of finite-sample noise, $\widehat{\mathcal{E}}_h(f_i, f^*)$ will not be exactly zero.

- We require **uniform concentration**: with high probability, the empirical error is close to the true error for *all* $g \in \mathcal{H}$.
- Let ϵ_{stat} be this uniform confidence radius.
- We relax the version space constraint to an elliptical radius R :

$$\sum_{i=0}^{t-1} \left(\widehat{\mathcal{E}}_h(f_i, g) \right)^2 \leq R^2 \quad \forall h$$

- We set $R^2 \approx t \cdot \epsilon_{\text{stat}}^2$ to ensure f^* safely remains in the version space.

Theorem (Sample-Based Optimistic Elimination)

Suppose $(\mathcal{M}, \mathcal{H})$ has Bellman Rank d and $f^* \in \mathcal{H}$. If we run Optimistic Elimination using the relaxed version space constraint:

- 1 **Iteration Complexity:** The algorithm terminates in $T = \tilde{O}(Hd)$ iterations.
- 2 **Sample Complexity:** With probability $\geq 1 - \delta$, the output policy is ϵ -optimal using a batch size per iteration of:

$$m = \tilde{O} \left(\frac{d^2 H^5}{\epsilon^2} \cdot \log \frac{|\mathcal{H}|}{\delta} \right)$$

Theorem (Sample-Based Optimistic Elimination)

Suppose $(\mathcal{M}, \mathcal{H})$ has Bellman Rank d and $f^* \in \mathcal{H}$. If we run Optimistic Elimination using the relaxed version space constraint:

- 1 **Iteration Complexity:** The algorithm terminates in $T = \tilde{O}(Hd)$ iterations.
- 2 **Sample Complexity:** With probability $\geq 1 - \delta$, the output policy is ϵ -optimal using a batch size per iteration of:

$$m = \tilde{O} \left(\frac{d^2 H^5}{\epsilon^2} \cdot \log \frac{|\mathcal{H}|}{\delta} \right)$$

Proof Gist: The exact dimension-counting proof is replaced by the **Elliptical Potential Lemma** (the log-determinant bound from LinUCB) applied to the cumulative errors.

Instantiations: What is the Sample Complexity?

Let us plug our two running examples into the general PAC bound.

Instantiations: What is the Sample Complexity?

Let us plug our two running examples into the general PAC bound.

1. Linear Bellman Completeness

- **Hypothesis Class:** All linear models $Q_h(s, a) = \phi(s, a)^\top w_h$.
- **Complexity:** Standard covering number arguments yield $\log |\mathcal{H}| \approx d \log(1/\epsilon)$.
- **Total Sample Complexity:** $\tilde{O}(d^3 H^6 / \epsilon^2)$.
- *Note: We achieve the same polynomial dependence on d as the LinMDP algorithm, but under a strictly weaker assumption.*

Instantiations: What is the Sample Complexity?

Let us plug our two running examples into the general PAC bound.

1. Linear Bellman Completeness

- **Hypothesis Class:** All linear models $Q_h(s, a) = \phi(s, a)^\top w_h$.
- **Complexity:** Standard covering number arguments yield $\log |\mathcal{H}| \approx d \log(1/\epsilon)$.
- **Total Sample Complexity:** $\tilde{O}(d^3 H^6 / \epsilon^2)$.
- *Note: We achieve the same polynomial dependence on d as the LinMDP algorithm, but under a strictly weaker assumption.*

2. Sparse Linear MDPs

- **Hypothesis Class:** Union of all s -sparse linear models out of D features.
- **Complexity:** $\log |\mathcal{H}| \approx s \log(eD/s) + s \log(1/\epsilon)$.
- **Total Sample Complexity:** $\tilde{O}(s^3 H^6 / \epsilon^2 \cdot \log(D))$.
- *Note: The statistical complexity is $\text{poly}(s, \log(D))$.*