Policy and Q-value Iteration: II

Gokul Swamy

(slides partially copied from Sanjiban / Wen;
all mistakes my own)

Recap

1. Policy Iteration and Q-value Iteration have monotonic improvement guarantees in the tabular setting, which we can prove via the PDL.

2. Scaling either method to larger problems requires function approximation for both policies and Q-functions.

3. However, these functions can be overly optimistic outside of their training distribution, leading to poor performance at test time (i.e. *distribution shift*).

Recap: Fitted Q-Iteration

Receive some dataset $\mathcal{D} = \{(s, a, r, s')\}$

Initialize $\hat{Q}_0 \in \mathcal{F}_Q, t \leftarrow 0$

for $t \in 1, \ldots, T$

$$\hat{Q}_{t+1} \leftarrow \arg\min_{Q \in \mathcal{F}_Q} \mathbb{E}_{\mathcal{D}}[(Q(s, a) - (r + \max_{a' \in \mathcal{A}} \hat{Q}_t(s', a')))^2]$$

Return π_T

Recap: Approximate Policy Iteration

Initialize with arbitrary π_0, $t = 0$

for $t \in 1, \ldots, T$

Sample $\mathcal{D}_t = \{(s_h, a_h, \hat{Q} = \sum_{\tau=h}^{H} r(s_\tau, a_\tau)) \sim \pi_t\}$

Fit $\hat{Q}_t \leftarrow \arg\min_{Q \in \mathcal{F}_Q} \mathbb{E}_{\mathcal{D}_t}[(Q(s,a) - \hat{Q})^2]$

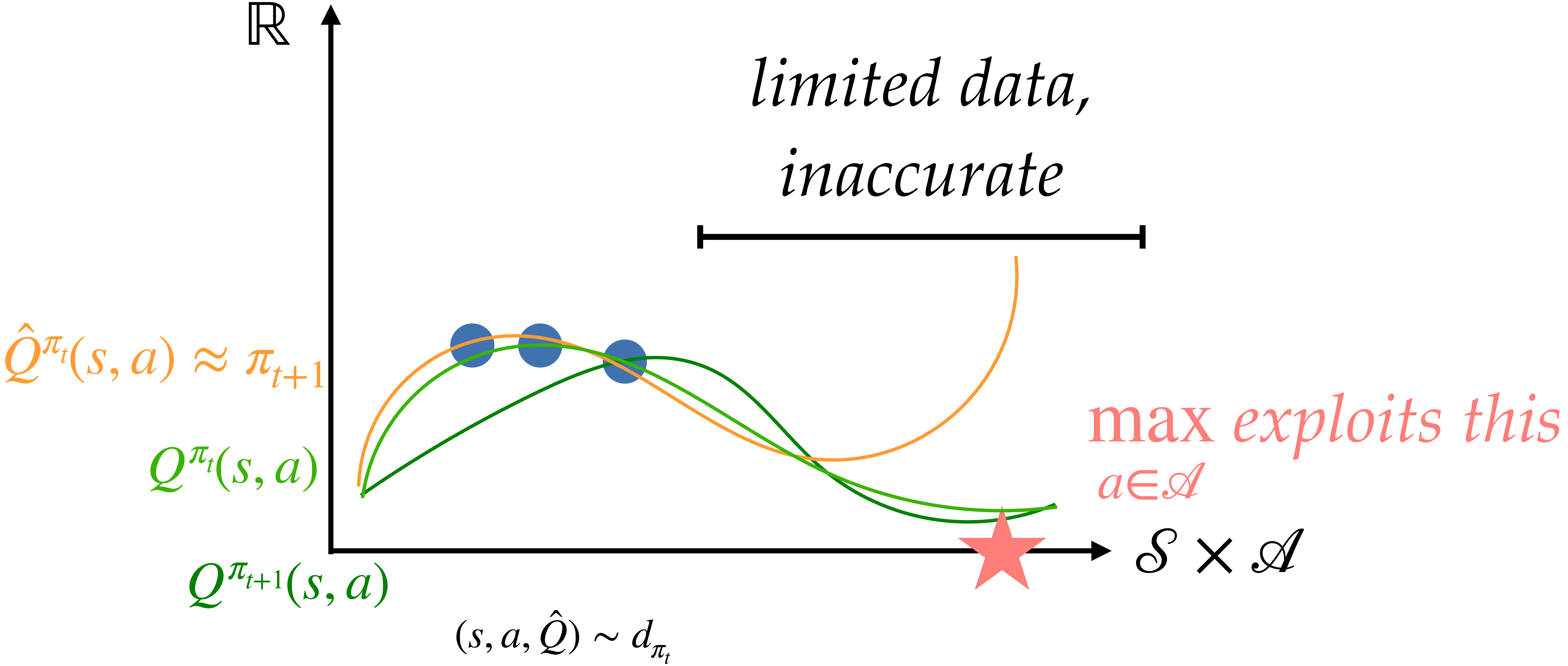
$\pi_{t+1}(s) = \arg\max_{a \in \mathcal{A}} \hat{Q}_t(s,a)$

if $\pi_{t+1} = \pi_t$: **break;**

Return π_T

Recap: Perils of FA in API

No More Monotonic Improvement! 😱😱😱



Outline for Today

1. Recap: Approximate Policy Iteration and Fitted Q-value Iteration

2. Diagnosing the Failures of Function Approximation in API

3. Cure 1: *Conservatism*

4. Cure 2: *Pessimism*

Outline for Today

1. Recap: Approximate Policy Iteration and Fitted Q-value Iteration

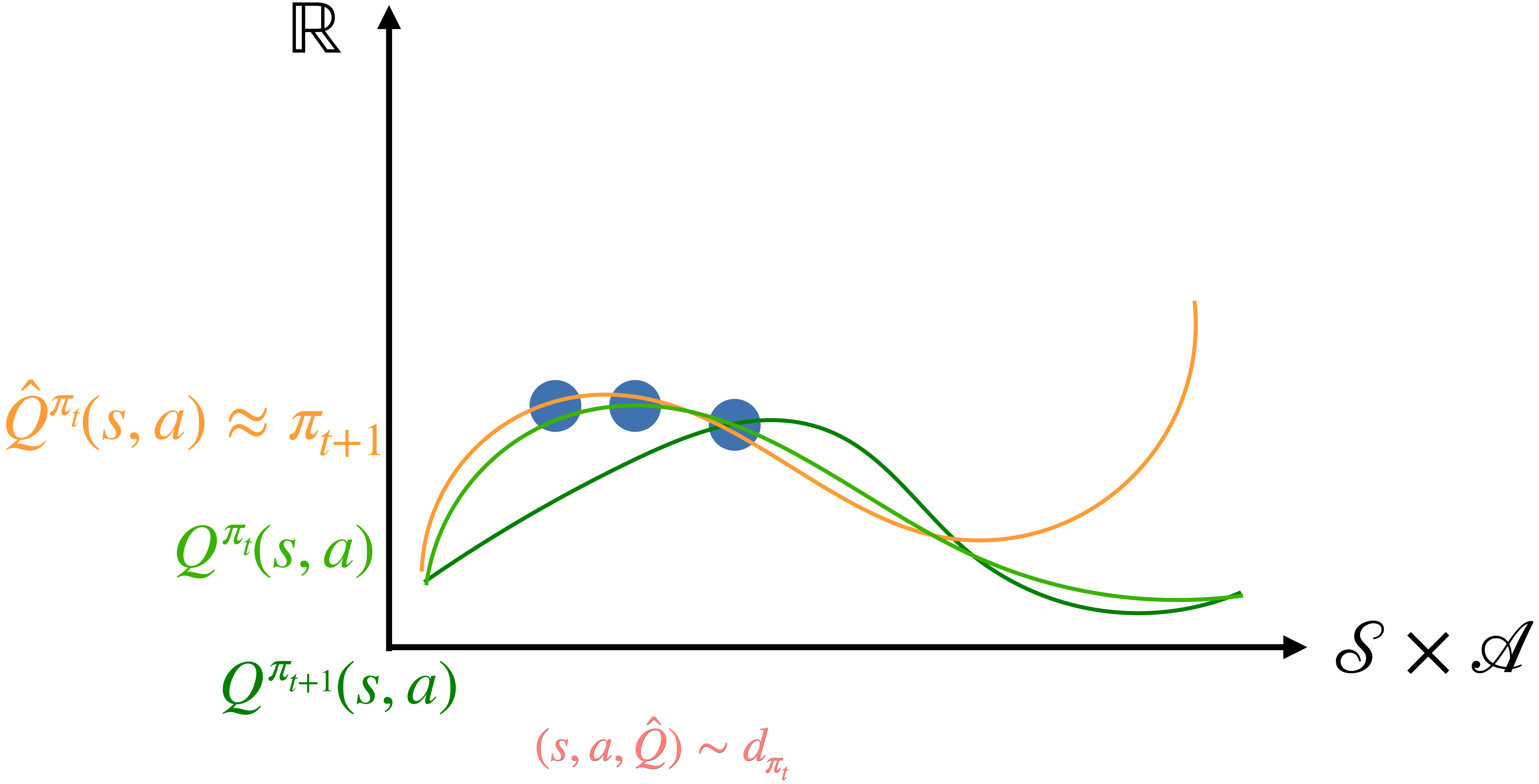
2. **Diagnosing the Failures of Function Approximation in API**

3. Cure 1: *Conservatism*

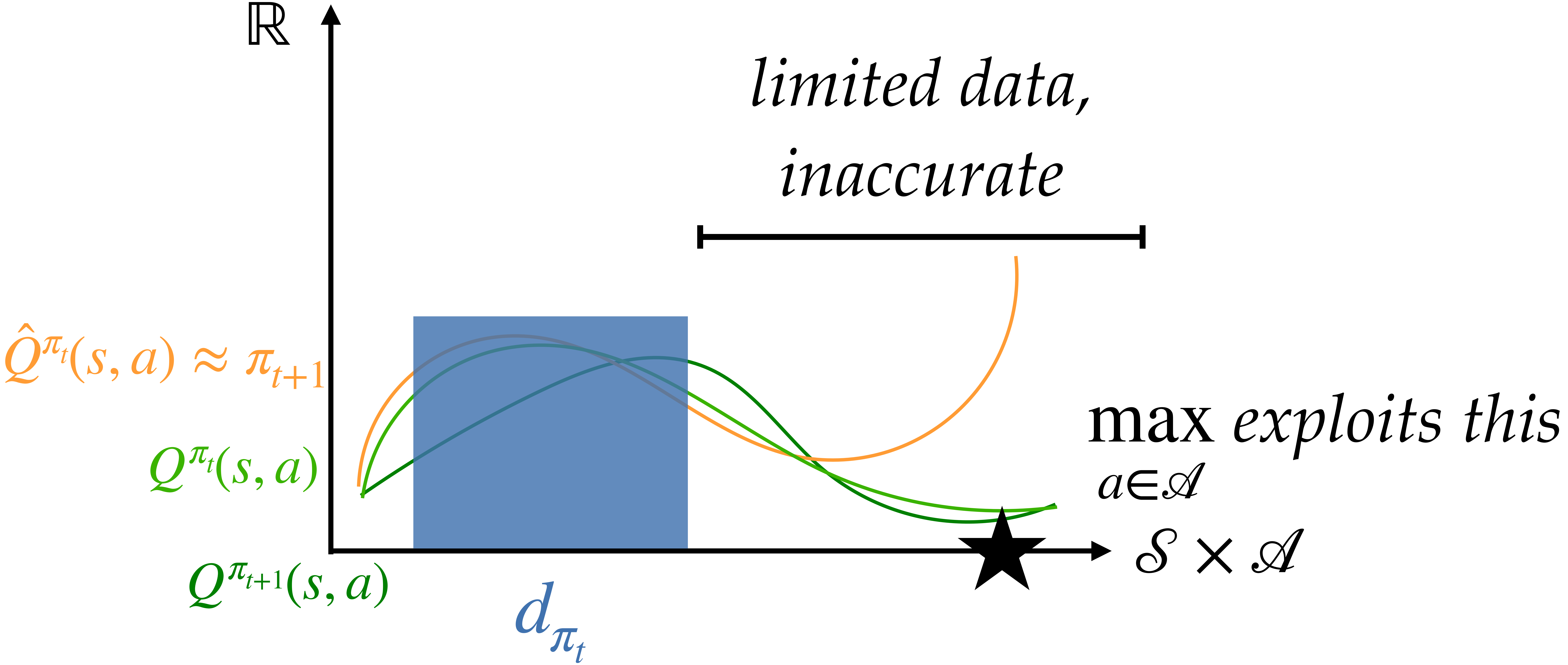
4. Cure 2: *Pessimism*

🔑 ***Q***: what "*broke*" in the preceding example?

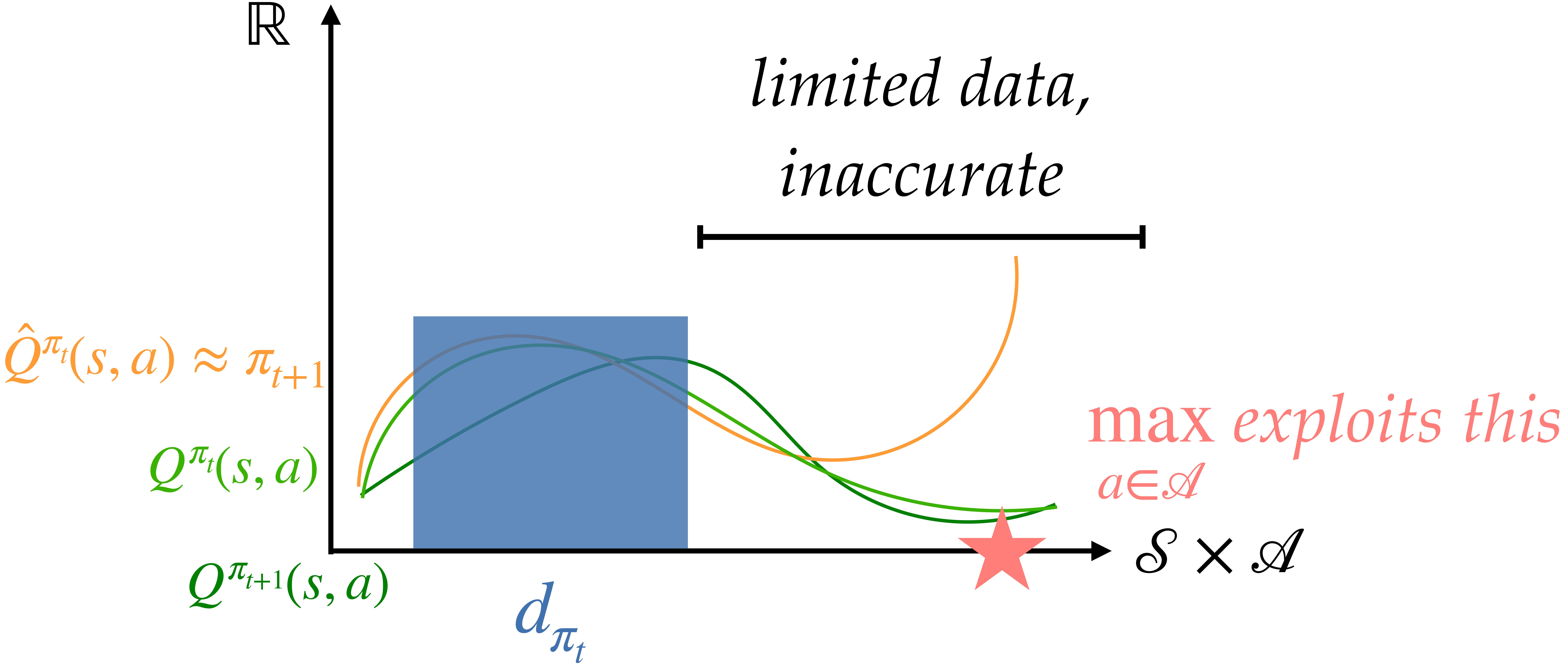
Diagnosis 1: Finite Samples!



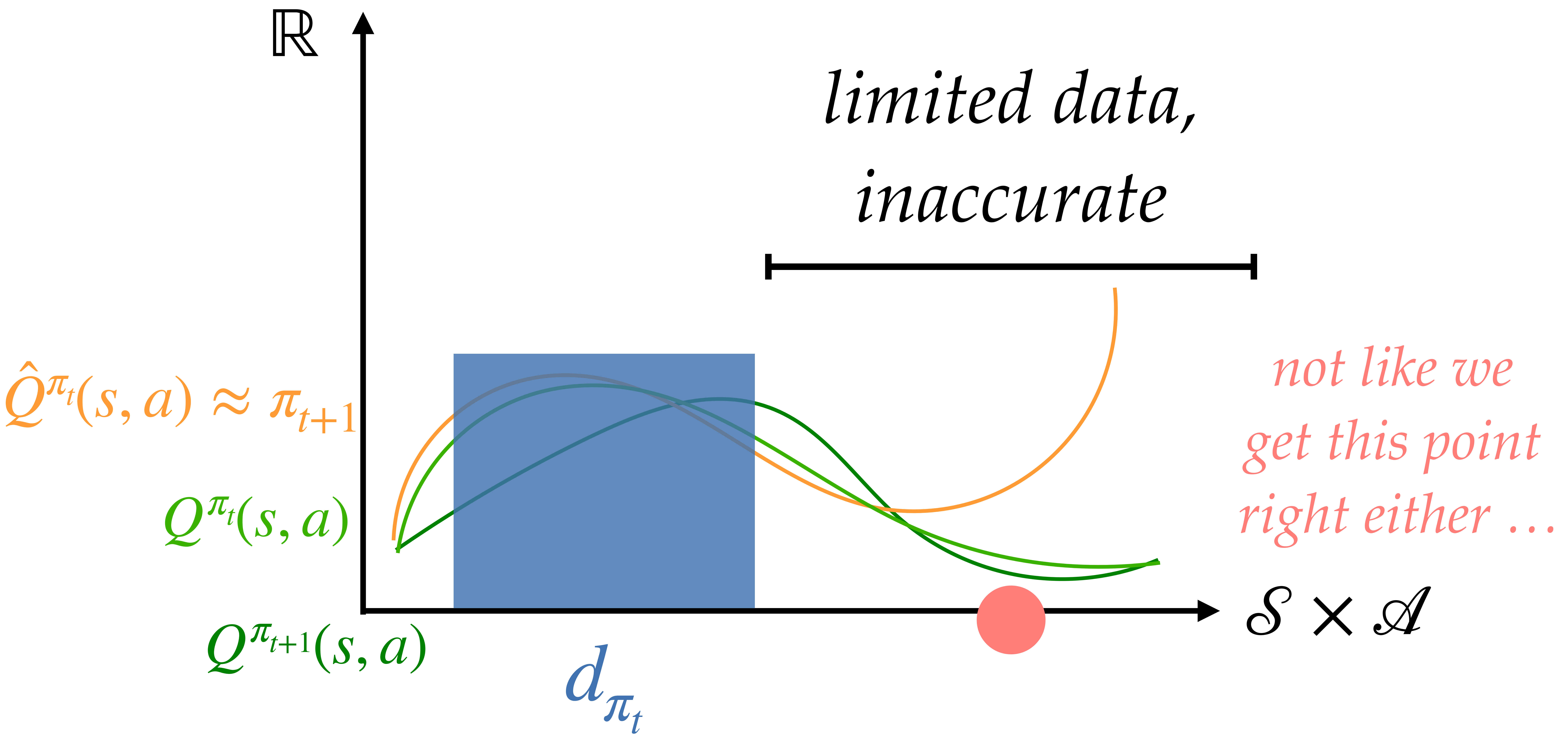
~~Diagnosis 1: Finite Samples!~~



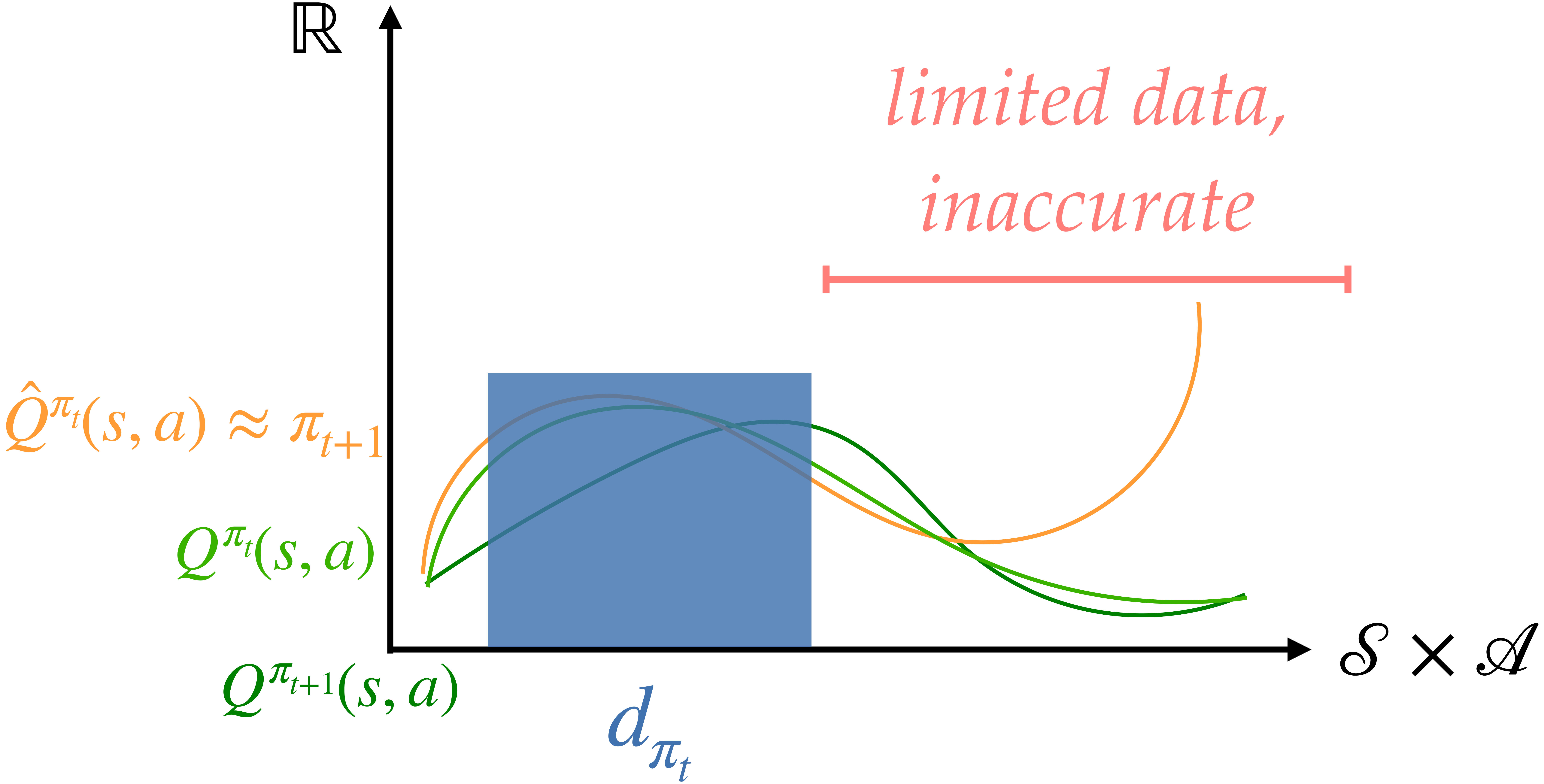
Diagnosis 2: The max!



Diagnosis 2: The max!



Diagnosis 3: Distribution Shift



Diagnosis 3: Distribution Shift

$$d_{\pi_t} \neq d_{\pi_{t+1}}$$

$$X = (s, a) \quad Y = \hat{Q}$$

$$P_{\text{train}}(X) \neq P_{\text{test}}(X)$$

Hard Q: what about the Y's?

Diagnosis 3: Distribution Shift

API: $\hat{Q}_t \leftarrow \arg\min_{Q \in \mathcal{F}_Q} \mathbb{E}_{\mathcal{D}_t}[(Q(s,a) - \hat{Q})^2]$

(on-policy — X's and Y's change)

FQI: $\hat{Q}_{t+1} \leftarrow \arg\min_{Q \in \mathcal{F}_Q} \mathbb{E}_{\mathcal{D}}[(Q(s,a) - (r + \max_{a' \in \mathcal{A}} \hat{Q}_t(s', a')))^2]$

(off-policy — X's change)

🔑 ***Q***: what “*broke*” in the preceding example?

A: distribution shift!


Sanjiban: Always has been.
Gokul: Wait it's just distribution shift???
imgflip.com

Outline for Today

1. Recap: Approximate Policy Iteration and Fitted Q-value Iteration

2. Diagnosing the Failures of Function Approximation in API

3. **Cure 1: *Conservatism***

4. Cure 2: *Pessimism*

The Performance Difference Lemma (PDL)

$$J(\pi_{t+1}) - J(\pi_t) = \mathbb{E}_{\xi \sim \pi_{t+1}} \left[\sum_h^H \underbrace{\mathbb{E}_{a' \sim \pi_{t+1}(s_h)}[Q^{\pi_t}(s_h, a')] - \mathbb{E}_{a' \sim \pi_t(s_h)}[Q^{\pi_t}(s_h, a')]}_{} \right]$$

exact PI: $\geq 0, \forall s \in \mathcal{S}$

🔑 ***Insight:*** we only need to be better on $d_{\pi_{t+1}}$, not $\forall s \in \mathcal{S}$!

Conservative Policy Iteration (CPI)

- Of course, we don't actually have access to $d_{\pi_{t+1}}$ before we actually compute π_{t+1}, giving us a chicken-and-egg problem.

- 🔑 ***Insight***: take a small step such that $d_{\pi_t} \approx d_{\pi_{t+1}}$. This means that our function approximators don't have to *extrapolate* and deal with too many OOD inputs!

Conservative Policy Iteration (CPI)

Initialize with arbitrary π_0, $t = 0$

for $t \in 1, \ldots, T$

Sample $\mathcal{D}_t = \{(s_h, a_h, \hat{Q} = \sum_{\tau=h}^{H} r(s_\tau, a_\tau)) \sim \pi_t\}$

Fit $\hat{Q}_t \leftarrow \arg\min_{Q \in \mathcal{F}_Q} \mathbb{E}_{\mathcal{D}_t}[(Q(s,a) - \hat{Q})^2]$

$\hat{\pi}_{t+1}(s) = \arg\max_{a \in \mathcal{A}} \hat{Q}_t(s,a)$, $\pi_{t+1} \leftarrow (1-\alpha)\pi_t + \alpha\hat{\pi}_{t+1}$

if $\pi_{t+1} = \pi_t$: **break;**

Return π_T

API is CPI with $\alpha = 1$!

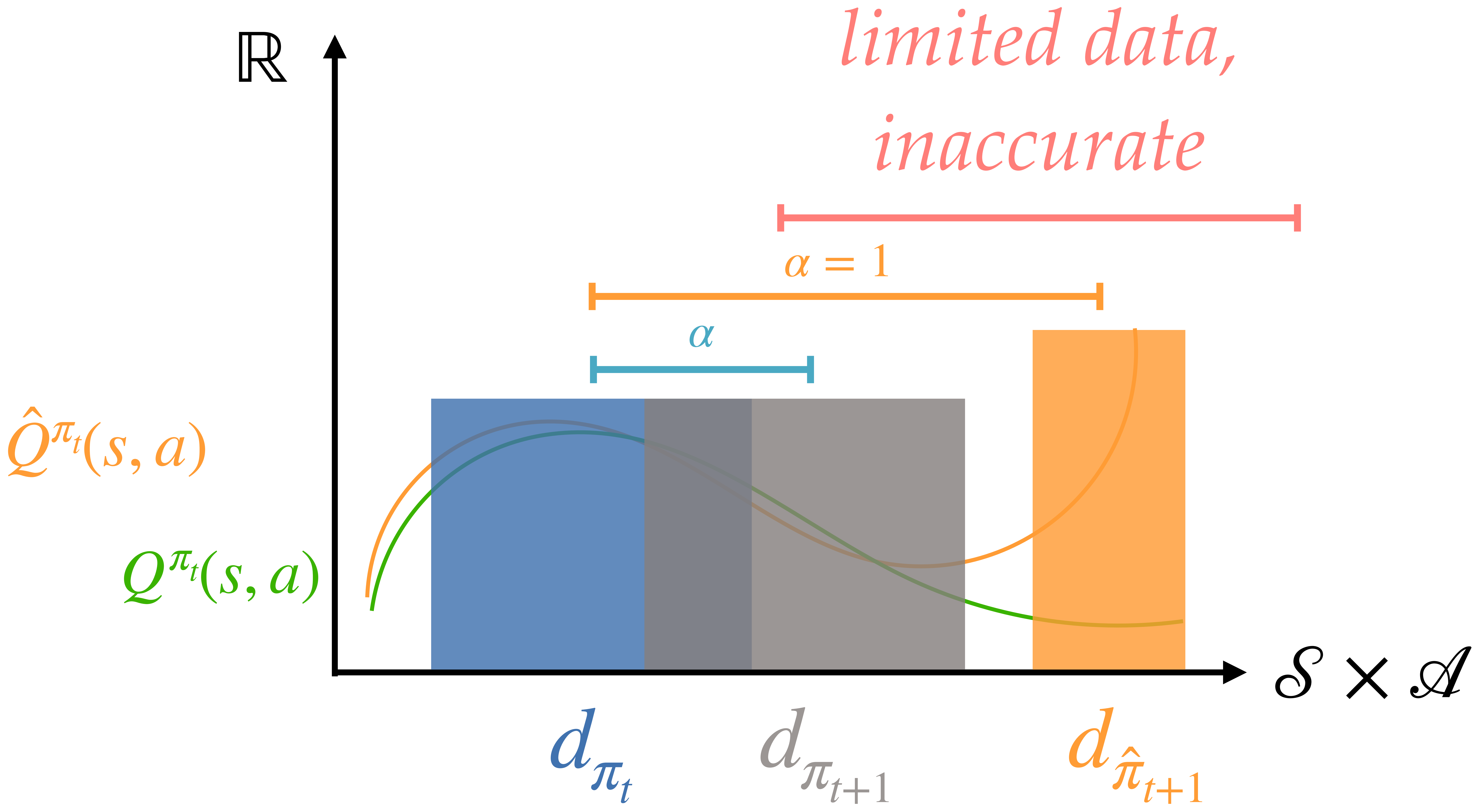
Conservative Policy Iteration (CPI)

- For a small enough α, we have $d_{\pi_t} \approx d_{\pi_{t+1}}$. Thus, if we mix in just the right amount of the *greedy policy* $\hat{\pi}_{t+1}$, we can guarantee monotonic improvement via PDL! See AJKS for proof.

- A different way of seeing this: API is CPI with $\alpha = 1$. This "learning rate" is too large to ensure we learn stably.

- Practical Deep RL algorithms like TRPO/PPO are built upon the conceptual bedrock of CPI.

Conservatism, Visually



How to take "small" steps?

- The update $\pi_{t+1} \leftarrow (1-\alpha)\pi_t + \alpha\hat{\pi}_{t+1}$ would require us to keep around a history of *all* greedy policies (i.e. all deep networks we've trained over iterations of the algorithm).

- Fundamentally, we're using this update to ensure that $d_{\pi_t} \approx d_{\pi_{t+1}}$ and our function approximators don't have to extrapolate too frequently.

- *Q:* can we implement the same principle in a different way?

How to take “small” steps?

$$\nabla_\theta J(\pi_\theta) = \sum_h^H \mathbb{E}_{s_h, a_h \sim \rho_h^{\pi_\theta}}[\nabla_\theta \log \pi_\theta(a_h \mid s_h) A^{\pi_\theta}(s_h, a_h)]$$

State-Action Distribution | *Likelihood Gradient* | *Advantage Function*

$$\theta_{t+1} \leftarrow \theta_t + \alpha \nabla_\theta J(\pi_{\theta_t})$$

How to take “small” steps?

$$\theta_{t+1} \leftarrow \theta_t + \alpha \nabla_\theta J(\pi_{\theta_t})$$

This is taking a small step in *parameter* space rather than in *policy* space — i.e. *not* the CPI principle!

$$\theta_{t+1} = \max_\theta \langle \nabla_\theta J(\pi_{\theta_t}), \theta - \theta_t \rangle$$

$$\text{s.t.} (\theta - \theta_t)^T (\theta - \theta_t) \leq \delta$$

How to take "small" steps?

$$\theta_{t+1} = \max_{\theta} \langle \nabla_\theta J(\pi_{\theta_t}), \theta - \theta_t \rangle$$

$$\text{s.t.} (\theta - \theta_t)^T (\theta - \theta_t) \leq \delta$$

🔑 Idea: We can update this constraint to enforce we are taking a small step in policy space!

How to take "small" steps?

$$\theta_{t+1} = \max_{\theta} \langle \nabla_\theta J(\pi_{\theta_t}), \theta - \theta_t \rangle$$

$$\text{s.t. } D_{KL}(d_{\theta_t} || d_\theta) \leq \delta$$

$$D_{KL}(d_{\theta_t} || d_\theta) \approx (\theta - \theta_t)^T F_{\theta_t} (\theta - \theta_t)$$

$$F_{\theta_t} = \mathbb{E}_{\pi_{\theta_t}}[\nabla_\theta \log \pi_{\theta_t}(a|s)^T \nabla_\theta \log \pi_{\theta_t}(a|s)]$$

🔑 Idea: We can update this constraint to enforce we are taking a small step in policy space!

How to take "small" steps?

$$\theta_{t+1} = \max_{\theta} \langle \nabla_\theta J(\pi_{\theta_t}), \theta - \theta_t \rangle$$

$$\text{s.t. } (\theta - \theta_t)^T F_{\theta_t} (\theta - \theta_t) \leq \delta$$

$$\Rightarrow \quad \theta_{t+1} \leftarrow \theta_t + \alpha F_{\theta_t}^{-1} \nabla_\theta J(\pi_{\theta_t})$$

🔑 Idea: The ***Natural*** Policy Gradient!

How to take “small” steps?

$$\nabla_\theta J(\pi_\theta) = \sum_h^H \mathbb{E}_{s_h, a_h \sim \rho_h^{\pi_\theta}}[\nabla_\theta \log \pi_\theta(a_h \mid s_h) A^{\pi_\theta}(s_h, a_h)]$$

State-Action Distribution *Likelihood Gradient* *Advantage Function*

$$F_{\theta_t} = \mathbb{E}_{\pi_{\theta_t}}[\nabla_\theta \log \pi_{\theta_t}(a \mid s)^T \nabla_\theta \log \pi_{\theta_t}(a \mid s)]$$

$$\theta_{t+1} \leftarrow \theta_t + \alpha F_{\theta_t}^{-1} \nabla_\theta J(\pi_{\theta_t})$$

PPO and TRPO are approximations of NPG!

Outline for Today

1. Recap: Approximate Policy Iteration and Fitted Q-value Iteration

2. Diagnosing the Failures of Function Approximation in API

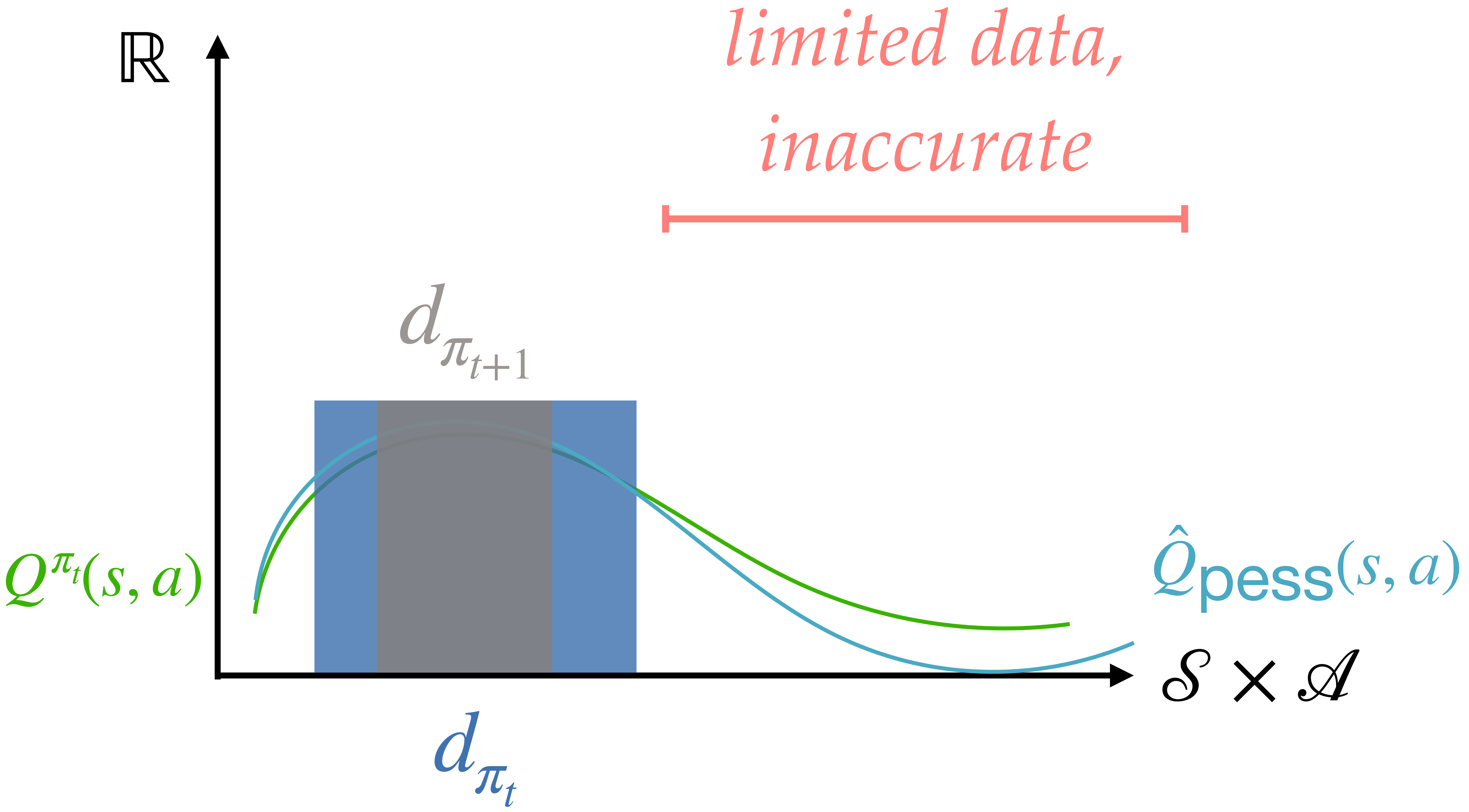
3. Cure 1: *Conservatism*

4. **Cure 2: *Pessimism***

Pessimism

- Conservatism is applying a step-size constraint in policy space, keeping the $\hat{Q}$-function fitting the same.
- Alternatively, we could change the way we approximate the $\hat{Q}$ to induce policies that don't go OOD.
- 🔑 ***Idea***: be as *pessimistic* as possible on unseen state/action pairs. The optimal policy under this $\hat{Q}$ won't want to ever leave the training distribution!
- We therefore don't need to iteratively collect new on-policy data via interaction. This is the core idea of *offline reinforcement learning*.

Pessimism, Visually



Summary

1. Function approximation causes problems in RL due to the shift between the training and the testing distributions.

2. We can address this problem in one of two ways:

   A. Take small steps (*conservatism*). We can do this by mixing or following the *natural policy gradient*.

   B. Assume the worst case thing happens OOD (*pessimism*).