Fitted Q-Iteration

(most references can be found on paper list for project topics)
Generalization for value-based batch RL

• We studied using abstractions to generalize in large state spaces
Generalization for value-based batch RL

• We studied using abstractions to generalize in large state spaces
• Abstractions correspond to “histogram regression” in supervised learning—the most trivial form of generalization
Generalization for value-based batch RL

• We studied using abstractions to generalize in large state spaces
• Abstractions correspond to “histogram regression” in supervised learning—the most trivial form of generalization
• Can I use XXX for value-based RL?
  • Linear predictors?
  • Kernal machines?
  • Random forests?
  • Neural nets???
  • ...

Generalization for value-based batch RL

• We studied using abstractions to generalize in large state spaces
• Abstractions correspond to “histogram regression” in supervised learning—the most trivial form of generalization
• Can I use XXX for value-based RL?
  • Linear predictors?
  • Kernal machines?
  • Random forests?
  • Neural nets??
  • …
• What you really want: *Reduction* of RL to supervised learning.
Generalization for value-based batch RL

- We studied using abstractions to generalize in large state spaces
- Abstractions correspond to “histogram regression” in supervised learning—the most trivial form of generalization
- Can I use XXX for value-based RL?
  - Linear predictors?
  - Kernal machines?
  - Random forests?
  - Neural nets???
  - …
- **What you really want:** Reduction of RL to supervised learning.
- And YES! We have already seen a special case of the algorithm…
Revisit MBRL (CE) with $\phi$

- Algorithm: estimate $\widehat{M}_\phi$, and do planning

$$\widehat{R}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} r, \quad \widehat{P}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} e_{\phi(s')}$$
Revisit MBRL (CE) with $\phi$

- Algorithm: estimate $\widehat{M}_\phi$, and do planning
  \[
  \widehat{R}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} r,
  \quad \widehat{P}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} e_{\phi(s')}
  \]

- Use Value Iteration as the planning algorithm:
Revisit MBRL (CE) with $\phi$

- Algorithm: estimate $\widehat{M}_\phi$, and do planning

$$\widehat{R}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} r, \quad \widehat{P}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} e_{\phi(s')}$$

- Use Value Iteration as the planning algorithm:
  - Initialize $g_0$ as any function in $\mathbb{R}^{|\phi(S)|}$
Revisit MBRL (CE) with $\phi$

- Algorithm: estimate $\widehat{M}_\phi$, and do planning:
  \[
  \widehat{R}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} r, \quad \widehat{P}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} \epsilon_{\phi(s')}
  \]

- Use Value Iteration as the planning algorithm:
  - Initialize $g_0$ as any function in $\mathbb{R}^{\phi(S)}$
  - $g_t \leftarrow \mathcal{T}_{\widehat{M}_\phi} g_{t-1}$. That is, for each $x \in \phi(S), a \in A$: 
Revisit MBRL (CE) with \( \phi \)

- Algorithm: estimate \( \hat{M}_\phi \), and do planning

\[
\hat{R}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} r, \quad \hat{P}_\phi(x, a) = \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} e_{\phi(s')}
\]

- Use Value Iteration as the planning algorithm:
  - Initialize \( g_0 \) as any function in \( \mathbb{R}^{|\phi(S)|} \)
  - \( g_t \leftarrow T_{\hat{M}_\phi} g_{t-1} \). That is, for each \( x \in \phi(S), a \in A \):

\[
g_t(x, a) = \hat{R}_\phi(x, a) + \gamma \langle \hat{P}_\phi(x, a), V_{g_{t-1}} \rangle \\
= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma \langle e_{\phi(s')}, V_{g_{t-1}} \rangle) \\
= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma V_{g_{t-1}}(\phi(s')))
\]
Rewrite in the original $S$

- Rewrite the algorithm so that $f_t = [g_t]_M$

\[
g_t(x, a) = \hat{R}_\phi(x, a) + \gamma \langle \hat{P}_\phi(x, a), V_{g_{t-1}} \rangle
\]

\[
= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma \langle e_{\phi(s')}, V_{g_{t-1}} \rangle)
\]

\[
= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma V_{g_{t-1}}(\phi(s')))
\]
Rewrite in the original $S$

- Rewrite the algorithm so that $f_t = [g_t]_M$
- Define $\mathcal{F}^\phi \subset \mathbb{R}^{|S \times A|}$ as the space of all functions over $S \times A$ that are piece-wise constant under $\phi$ with value in $[0, \frac{R_{\text{max}}}{1-\gamma}]$

\[
g_t(x, a) = \hat{R}_\phi(x, a) + \gamma \langle \hat{P}_\phi(x, a), V_{g_{t-1}} \rangle
\]
\[
= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma \langle e_{\phi(s')}, V_{g_{t-1}} \rangle)
\]
\[
= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma V_{g_{t-1}}(\phi(s')))
\]
Rewrite in the original $S$

- Rewrite the algorithm so that $f_t = [g_t]_M$
- Define $\mathcal{F}^\phi \subset \mathbb{R}^{|S \times A|}$ as the space of all functions over $S \times A$ that are piece-wise constant under $\phi$ with value in $[0, \frac{R_{\text{max}}}{1-\gamma}]$
- Initialize $f_0$ as any function in $F^\phi$

$$g_t(x, a) = \hat{R}_\phi(x, a) + \gamma \langle \hat{P}_\phi(x, a), V_{g_{t-1}} \rangle$$

$$= \frac{1}{|D_{x, a}|} \sum_{(r, s') \in D_{x, a}} (r + \gamma \langle e_\phi(s'), V_{g_{t-1}} \rangle)$$

$$= \frac{1}{|D_{x, a}|} \sum_{(r, s') \in D_{x, a}} (r + \gamma V_{g_{t-1}}(\phi(s')))$$
Rewrite in the original $S$

- Rewrite the algorithm so that $f_t = [g_t]_M$
- Define $\mathcal{F}^\phi \subset \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$ as the space of all functions over $\mathcal{S} \times \mathcal{A}$ that are piece-wise constant under $\phi$ with value in $[0, R_{\text{max}}/(1-\gamma)]$
- Initialize $f_0$ as any function in $\mathcal{F}^\phi$
- For each $s \in \mathcal{S}, a \in \mathcal{A}$:

\[
\begin{align*}
  f_t(s, a) &= \hat{R}_\phi(\phi(s), a) + \gamma \langle \hat{P}_\phi(\phi(s), a), [V_{f_{t-1}}]_\phi \rangle \\
  &= \frac{1}{|D_{\phi(s),a}|} \sum_{(r,s') \in D_{\phi(s),a}} (r + \gamma \langle e_\phi(s'), [V_{f_{t-1}}]_\phi \rangle) \\
  &= \frac{1}{|D_{\phi(s),a}|} \sum_{(r,s') \in D_{\phi(s),a}} (r + \gamma V_{f_{t-1}}(s'))
\end{align*}
\]

\[
\begin{align*}
  g_t(x, a) &= \hat{R}_\phi(x, a) + \gamma \langle \hat{P}_\phi(x, a), [V_{g_{t-1}}] \rangle \\
  &= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma \langle e_\phi(s'), [V_{g_{t-1}}]_\phi \rangle) \\
  &= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma V_{g_{t-1}}(\phi(s'))) \\
\end{align*}
\]
Rewrite in the original $S$

- Rewrite the algorithm so that $f_t = [g_t]_M$
- Define $\mathcal{F}_\phi \subset \mathbb{R}^{|S \times A|}$ as the space of all functions over $S \times A$ that are piece-wise constant under $\phi$ with value in $[0, R_{\text{max}}/(1-\gamma)]$
- Initialize $f_0$ as any function in $F^\phi$
- For each $s \in S$, $a \in A$: essentially $f_t \leftarrow \mathcal{T}_{\hat{M}_\phi} f_{t-1}$

$$
\begin{align*}
    f_t(s,a) &= \hat{R}_\phi(\phi(s),a) + \gamma \langle \hat{P}_\phi(\phi(s),a), [V_{f_{t-1}}]_\phi \rangle \\
    &= \frac{1}{|D_{\phi(s),a}|} \sum_{(r,s') \in D_{\phi(s),a}} (r + \gamma \langle e_{\phi(s')}, [V_{f_{t-1}}]_\phi \rangle) \\
    &= \frac{1}{|D_{\phi(s),a}|} \sum_{(r,s') \in D_{\phi(s),a}} (r + \gamma V_{f_{t-1}}(s')) \\
    g_t(x,a) &= \hat{R}_\phi(x,a) + \gamma \langle \hat{P}_\phi(x,a), V_{g_{t-1}} \rangle \\
    &= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma \langle e_{\phi(s')}, V_{g_{t-1}} \rangle) \\
    &= \frac{1}{|D_{x,a}|} \sum_{(r,s') \in D_{x,a}} (r + \gamma V_{g_{t-1}}(\phi(s')))
\end{align*}
$$
Rewrite in the original $S$

- Rewrite the algorithm so that $f_t = [g_t]_M$
- Define $\mathcal{F}^\phi \subset \mathbb{R}^{|S \times A|}$ as the space of all functions over $S \times A$ that are piece-wise constant under $\phi$ with value in $[0, R_{\text{max}}/(1-\gamma)]$
- Initialize $f_0$ as any function in $F^\phi$
- For each $s \in S, a \in A$: essentially $f_t \leftarrow \mathcal{T}_{\hat{\mathcal{M}}_{\phi}} f_{t-1}$

$$ f_t(s, a) = \hat{R}_\phi(\phi(s), a) + \gamma \langle \hat{P}_\phi(\phi(s), a), [V_{f_{t-1}}]_\phi \rangle \quad g_t(x, a) = \hat{R}_\phi(x, a) + \gamma \langle \hat{P}_\phi(x, a), V_{g_{t-1}} \rangle $$

$$ = \frac{1}{|D_{\phi(s), a}|} \sum_{(r, s') \in D_{\phi(s), a}} \left( r + \gamma \langle e_{\phi(s')}, [V_{f_{t-1}}]_\phi \rangle \right) $$

$$ = \frac{1}{|D_{x, a}|} \sum_{(r, s') \in D_{x, a}} \left( r + \gamma \langle e_{\phi(s')}, V_{g_{t-1}} \rangle \right) $$

"Empirical Bellman update"
(based on 1 data point)
Alternative interpretation of the above step

\[ f_t(s, a) = \frac{1}{|D_{\phi(s), a}|} \sum_{(r, s') \in D_{\phi(s), a}} \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \]
Alternative interpretation of the above step

• Dataset $D = \{(s, a, r, s')\}$

\[
f_t(s, a) = \frac{1}{|D_{\phi(s), a}|} \sum_{(r, s') \in D_{\phi(s), a}} \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right)
\]
Alternative interpretation of the above step

- Dataset \( D = \{ (s, a, r, s') \} \)
- Apply emp. Bellman up. to \( f_{t-1} \) based on each data point:
  \[
  f_t(s, a) = \frac{1}{|D_{\phi(s), a}|} \sum_{(r, s') \in D_{\phi(s), a}} \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right)
  \]
Alternative interpretation of the above step

• Dataset \( D = \{(s, a, r, s')\} \)

• Apply emp. Bellman up. to \( f_{t-1} \) based on each data point:

\[
\left\{ \left( (s, a), (r + \gamma \max_{a' \in A} f_{t-1}(s', a')) \right) \right\}
\]

• What does it mean to take average over \( D_{\phi(s), a} \)?
\[ f_t(s, a) = \frac{1}{|D_{\phi(s), a}|} \sum_{(r, s', a') \in D_{\phi(s), a}} \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \]

Alternative interpretation of the above step

• Dataset \( D = \{(s, a, r, s')\} \)

• Apply emp. Bellman up. to \( f_{t-1} \) based on each data point:
  \[ \left\{ \left( (s, a), (r + \gamma \max_{a' \in A} f_{t-1}(s', a')) \right) \right\} \]

• What does it mean to take average over \( D_{\phi(s), a} \)?
  • Recall: average minimizes mean squared error (MSE)
\[ f_t(s, a) = \frac{1}{|\mathcal{D}_\phi(s, a)|} \sum_{(r, s') \in \mathcal{D}_\phi(s, a)} \left( r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a') \right) \]

Alternative interpretation of the above step

- Dataset \( D = \{(s, a, r, s')\} \)
- Apply emp. Bellman up. to \( f_{t-1} \) based on each data point:
  \[ \left\{ \left( (s, a), (r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a')) \right) \right\} \]
- What does it mean to take average over \( D_{\phi(s), a} \)?
  - Recall: average minimizes mean squared error (MSE)
  - *Projection* onto \( F^\phi \)! (think of functions over \( D \))

\[ f_t = \arg \min_{f \in F^\phi} \sum_{(s, a, r, s') \in D} \left( f(s, a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a') \right) \right)^2 \]
Alternative interpretation of the above step

• Dataset \( D = \{(s, a, r, s')\} \)

• Apply emp. Bellman up. to \( f_{t-1} \) based on each data point:

\[
\left\{ \left( (s, a), \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right) \right\}
\]

• What does it mean to take average over \( D_{\phi(s,a)} \)?

  • Recall: average minimizes mean squared error (MSE)
  • \textit{Projection} onto \( F^\phi \) ! (think of functions over \( D \))

\[
f_t = \arg \min_{f \in F^\phi} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2
\]

• … which is, solving a SL regression problem with histogram regression \( F^\phi \)
Alternative interpretation of the above step

- Dataset $D = \{(s, a, r, s')\}$
- Apply emp. Bellman up. to $f_{t-1}$ based on each data point:

$$\left\{ \left( (s, a), (r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a')) \right) \right\}$$

- What does it mean to take average over $D_{\phi(s), a}$?

  - Recall: average minimizes mean squared error (MSE)
  - Projection onto $F^\phi$! (think of functions over $D$)

$$f_t = \arg \min_{f \in F^\phi} \sum_{(s, a, r, s') \in D} \left( f(s, a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a') \right) \right)^2$$

- … which is, solving a SL regression problem with histogram regression $F^\phi$
- Reduction done! Plug in your favorite SL method
Fitted Q-Iteration (FQI): \[ f_t = \arg \min_{f \in F} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2 \]

[Ernst et al'05]; see also [Gordon'95]
Fitted Q-Iteration (FQI):  \( f_t = \arg\min_{f \in \mathcal{F}} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a') \right) \right)^2 \)

[Ernst et al'05]; see also [Gordon'95]

We simplified a “regression algorithm” to its corresponding function space \( \mathcal{F} \)
Fitted Q-Iteration (FQI):  
\[ f_t = \arg \min_{f \in F} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2 \]

[Ernst et al'05]; see also [Gordon'95]

We simplified a “regression algorithm” to its corresponding function space \( F \)

- Empirical Risk Minimization (ERM); assume optimization is exact; does not consider regularization, etc.
We simplified a “regression algorithm” to its corresponding function space $F$

- Empirical Risk Minimization (ERM); assume optimization is exact; does not consider regularization, etc.
- Will also assume finite (but exponentially large) $F$

Fitted Q-Iteration (FQI): $f_t = \arg \min_{f \in F} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s',a') \right) \right)^2$

[Ernst et al’05]; see also [Gordon’95]
We simplified a “regression algorithm” to its corresponding function space $F$

- **Empirical Risk Minimization (ERM);** assume optimization is exact; does not consider regularization, etc.
- Will also assume finite (but exponentially large) $F$
  - continuous spaces are often handled by discretization in SLT (e.g., growth function, covering number)

Fitted Q-Iteration (FQI): \[ f_t = \arg \min_{f \in F} \sum_{(s, a, r, s') \in D} \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2 \]

[Ernst et al’05]; see also [Gordon’95]
We simplified a “regression algorithm” to its corresponding function space $F$

- Empirical Risk Minimization (ERM); assume optimization is exact; does not consider regularization, etc.
- Will also assume finite (but exponentially large) $F$
  - continuous spaces are often handled by discretization in SLT (e.g., growth function, covering number)
  - methods like regression trees have dynamic function spaces (and often need SRM); not accommodated

Fitted Q-Iteration (FQI): $f_t = \arg \min_{f \in \mathcal{F}} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2$

[Ernst et al’05]; see also [Gordon’95]
We simplified a “regression algorithm” to its corresponding function space $F$

- Empirical Risk Minimization (ERM); assume optimization is exact; does not consider regularization, etc.
- Will also assume finite (but exponentially large) $F$
  - continuous spaces are often handled by discretization in SLT (e.g., growth function, covering number)
  - methods like regression trees have dynamic function spaces (and often need SRM); not accommodated
- A minimal but (hopefully) insightful simplification of supervised learning

Fitted Q-Iteration (FQI): $f_t = \arg\min_{f \in F} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s',a') \right) \right)^2$
Fitted Q-Iteration (FQI): 
\[ f_t = \arg \min_{f \in \mathcal{F}} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2 \]

Asynchronous update + stochastic approximation?
Fitted Q-Iteration (FQI): \( f_t = \arg \min_{f \in \mathcal{F}} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s',a') \right) \right)^2 \)

Asynchronous update + stochastic approximation?

- Assume parameterized & differentiable function: \( \mathcal{F} = \{ f_{\theta} : \theta \in \Theta \} \)
Fitted Q-Iteration (FQI): \[ f_t = \arg\min_{f \in \mathcal{F}} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s',a') \right) \right)^2 \]

Asynchronous update + stochastic approximation?

- Assume parameterized & differentiable function: \( \mathcal{F} = \{ f_\theta : \theta \in \Theta \} \)
- Online regression: randomly pick a data point and do a stochastic gradient update:

\[
\theta \leftarrow \theta - \frac{\alpha}{2} \cdot \nabla_\theta \left( f_\theta(s,a) - \left( r + \gamma \max_{a' \in A} f_\theta(s',a') \right) \right)^2
\]

\[
= \theta - \alpha \left( f_\theta(s,a) - \left( r + \gamma \max_{a' \in A} f_\theta(s',a') \right) \right) \nabla_\theta f_\theta(s,a)
\]

Treat as constant; don’t pass gradient
Asynchronous update + stochastic approximation?

- Assume parameterized & differentiable function: \( \mathcal{F} = \{ f_\theta : \theta \in \Theta \} \)
- Online regression: randomly pick a data point and do a stochastic gradient update:

\[
\theta \leftarrow \theta - \frac{\alpha}{2} \cdot \nabla_{\theta} \left( f_\theta(s, a) - \left( r + \gamma \max_{a' \in A} f_{\theta}(s', a') \right) \right)^2
\]

\[
= \theta - \alpha \left( f_\theta(s, a) - \left( r + \gamma \max_{a' \in A} f_{\theta}(s', a') \right) \right) \nabla_{\theta} f_\theta(s, a)
\]

- If \( f_\theta \) is the tabular function, it’s (tabular) Q-learning

Fitted Q-Iteration (FQI):

\[
f_t = \arg \min_{f \in \mathcal{F}} \sum_{(s, a, r, s') \in D} \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2
\]

[Ernst et al'05]; see also [Gordon'95]
Asynchronous update + stochastic approximation?

- Assume parameterized & differentiable function: \( \mathcal{F} = \{ f_\theta : \theta \in \Theta \} \)
- Online regression: randomly pick a data point and do a stochastic gradient update:
  \[
  \theta \leftarrow \theta - \alpha \cdot \nabla_\theta \left( f_\theta(s, a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2
  = \theta - \alpha \left( f_\theta(s, a) - \left( r + \gamma \max_{a' \in A} f_{\theta}(s', a') \right) \right) \nabla_\theta f_\theta(s, a)
  \]
- If \( f_\theta \) is the tabular function, it’s (tabular) Q-learning
- If \( f_\theta \) is a neural net, it’s (almost) DQN (Mnih et al.’15)
Asynchronous update + stochastic approximation?

- Assume parameterized & differentiable function: \( \mathcal{F} = \{ f_\theta : \theta \in \Theta \} \)
- Online regression: randomly pick a data point and do a stochastic gradient update:
  \[
  \theta \leftarrow \theta - \frac{\alpha}{2} \cdot \nabla_\theta \left( f_\theta(s, a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2
  = \theta - \alpha \left( f_\theta(s, a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right) \nabla_\theta f_\theta(s, a)
  \]

- If \( f_\theta \) is the tabular function, it’s (tabular) Q-learning
- If \( f_\theta \) is a neural net, it’s (almost) DQN (Mnih et al.’15)
  - Using a target network is even more similar to FQI

Fitted Q-Iteration (FQI):

\[
f_t = \arg \min_{f \in \mathcal{F}} \sum_{(s, a, r, s') \in D} \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right)^2
\]

[Ernst et al’05; see also [Gordon’95]
Fitted Q-Iteration (FQI): \[ f_t = \arg \min_{f \in \mathcal{F}} \sum_{(s,a,r,s') \in D} \left( f(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f_{t-1}(s', a') \right) \right)^2 \]

Each minimization step plays two roles:

1. Denoise the emp update \( r + \gamma V_f(s') \) to \((\mathcal{T}f)(s, a)\) (w/ inf data)
   - This happens even in tabular setting

2. \( \mathcal{T}f \) may not be manageable => find the closest approximation in \( F \) (i.e., projection)
   - Will use \( \prod F \) as the projection. Dependence on weights over state-action pairs omitted—determined by data distribution
   - With infinite data, FQI becomes: \( f_t \leftarrow \prod F \mathcal{T}f_{t-1} \)
Convergence?

• First: with infinite data, $Q^*$ is a fixed point (as long as $Q^* \in F$)
Convergence?

- First: with infinite data, $Q^*$ is a fixed point (as long as $Q^* \in F$)
- We know: CE w/ $Q^*$-irrelevant abstractions is a special case of FQI—convergence guaranteed.
Convergence?

- First: with infinite data, $Q^*$ is a fixed point (as long as $Q^* \in F$)
- We know: CE w/ $Q^*$-irrelevant abstractions is a special case of FQI—convergence guaranteed.
- But: not necessarily with general function spaces! Can diverge even with linear functions, under realizability (i.e., $Q^* \in F$)
2.1 Counter-example for least-square regression [Tsitsiklis and van Roy, 1996]

An MDP with two states $x_1, x_2$, 1-d features for the two states: $f_{x_1} = 1$, $f_{x_2} = 2$. Linear Function approximation with $\hat{V}_\theta(x) = \theta f_x$.

$$\theta_k := \arg \min_\theta \frac{1}{2} (\theta - \text{target}_1)^2 + (2\theta - \text{target}_2)^2$$

$$= \arg \min_\theta \frac{1}{2} (\theta - \gamma \theta^{k-1} f_{x_2})^2 + (2\theta - \gamma \theta^{k-1} f_{x_2})^2$$

$$= \arg \min_\theta \frac{1}{2} (\theta - \gamma 2\theta^{k-1})^2 + (2\theta - \gamma 2\theta^{k-1})^2$$

$$(\theta - \gamma 2\theta^{k-1}) + 2(2\theta - \gamma 2\theta^{k-1}) = 0 \Rightarrow 5\theta = 6\gamma \theta^{k-1}$$

$$\therefore \theta_k = \frac{6}{5} \gamma \theta_{k-1}$$

This diverges if $\gamma \geq 5/6$. 

credit: course notes from Shipra Agrawal
Bellman error minimization

- Standard VI: $f_t \leftarrow \mathcal{I} f_{t-1}$
Bellman error minimization

- Standard VI: $f_t \leftarrow \mathcal{I} f_{t-1}$
- FQI keeps things tractable by: $f_t \leftarrow \prod_F \mathcal{I} f_{t-1}$
Bellman error minimization

- Standard VI: \( f_t \leftarrow \mathcal{I} f_{t-1} \)
- FQI keeps things tractable by: \( f_t \leftarrow \prod_F \mathcal{I} f_{t-1} \)
  - We know \( \mathcal{I} \) is \( \gamma \)-contraction.
Bellman error minimization

• Standard VI: \( f_t \leftarrow \mathcal{T} f_{t-1} \)

• FQI keeps things tractable by: \( f_t \leftarrow \prod F \mathcal{T} f_{t-1} \)
  
  • We know \( \mathcal{T} \) is \( \gamma \)-contraction.
  
  • If \( \prod F \) is non-expansion, we are good — not always true though
    (You should have seen an example of non-expansion \( \prod F \). Which one?)
Bellman error minimization

- Standard VI: \( f_t \leftarrow \mathcal{T} f_{t-1} \)
- FQI keeps things tractable by: \( f_t \leftarrow \prod_F \mathcal{T} f_{t-1} \)
  - We know \( \mathcal{T} \) is \( \gamma \)-contraction.
  - If \( \prod_F \) is non-expansion, we are good — not always true though (You should have seen an example of non-expansion \( \prod_F \). Which one?)
  - Still an iterated algorithm for fixed point eq. \( \Rightarrow \) no globally consistent optimization objective! (objective changes as current \( f \) changes)
Bellman error minimization

- Standard VI: \( f_t \leftarrow \mathcal{T} f_{t-1} \)
- FQI keeps things tractable by: \( f_t \leftarrow \prod_F \mathcal{T} f_{t-1} \)
  - We know \( \mathcal{T} \) is \( \gamma \)-contraction.
  - If \( \prod_F \) is non-expansion, we are good — not always true though (You should have seen an example of non-expansion \( \prod_F \). Which one?)
  - Still an iterated algorithm for fixed point eq. => no globally consistent optimization objective! (objective changes as current \( f \) changes)
- Alternative: minimize \( \| f - \mathcal{T} f \| \) over \( f \in F \)
Bellman error minimization

- Standard VI: \( f_t \leftarrow T f_{t-1} \)
- FQI keeps things tractable by: \( f_t \leftarrow \prod_F T f_{t-1} \)
  - We know \( T \) is \( \gamma \)-contraction.
  - If \( \prod_F \) is non-expansion, we are good — not always true though (You should have seen an example of non-expansion \( \prod_F \). Which one?)
  - Still an iterated algorithm for fixed point eq. \( \Rightarrow \) no globally consistent optimization objective! (objective changes as current \( f \) changes)
- Alternative: minimize \( \| f - T f \| \) over \( f \in F \)
  - Is it equivalent to minimizing: \( E_{(s,a)\sim\nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a'\in A} f(s', a') \right) \right)^2 \right] \)?
Bellman error minimization

- Standard VI: \( f_t \leftarrow T f_{t-1} \)
- FQI keeps things tractable by: \( f_t \leftarrow \prod_F T f_{t-1} \)
  - We know \( T \) is \( \gamma \)-contraction.
  - If \( \prod_F \) is non-expansion, we are good — not always true though (You should have seen an example of non-expansion \( \prod_F \). Which one?)
  - Still an iterated algorithm for fixed point eq. \( \Rightarrow \) no globally consistent optimization objective! (objective changes as current \( f \) changes)
- Alternative: minimize \( || f - T f || \) over \( f \in F \)
  - Is it equivalent to minimizing: 
    \[
    \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right] ?
    \] (omitted in the rest of slides)
Bellman error minimization

$$E_{(s,a) \sim \nu} \left[ \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right]$$
Bellman error minimization

\[ E_{(s,a)\sim \nu} \left[ \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right] \]

\[ = E_{(s,a)\sim \nu} \left[ (f(s, a) - (Tf)(s, a))^2 \right] + E_{(s,a)\sim \nu} \left[ \left( (Tf)(s, a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right] \]
Bellman error minimization

\begin{equation}
\mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right] = \mathbb{E}_{(s,a) \sim \nu} \left[ (f(s,a) - (\mathcal{T} f)(s,a))^2 \right] + \mathbb{E}_{(s,a) \sim \nu} \left[ (\mathcal{T} f)(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right]^2
\end{equation}

This part is what we want: \( \| f - \mathcal{T} f \| \), with a weighted 2-norm defined w/ \( \nu \)
Bellman error minimization

$$
E_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right]
$$

$$
= E_{(s,a) \sim \nu} \left[ (f(s,a) - (Tf)(s,a))^2 \right] + E_{(s,a) \sim \nu} \left[ \left( (Tf)(s,a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right]
$$

**This part is what we want:**
\[\| f - T f \|, \text{ with a weighted 2-norm defined w/ } \nu\]

**This part is annoying!**

- Prefer “flat” \( f \)
- \( Q^* \) is not necessarily flat!
- 0 for deterministic transitions. Issue is only serious when env highly stochastic.
Bellman error minimization

\[
\mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right]
\]

\[
= \mathbb{E}_{(s,a) \sim \nu} \left[ (f(s,a) - (Tf)(s,a))^2 \right] + \mathbb{E}_{(s,a) \sim \nu} \left[ (Tf)(s,a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right)^2 \right]
\]

This part is what we want:
\[ \| f - T f \| , \text{ with a weighted 2-norm defined w/ } \nu \]

This part is annoying!

- Prefer “flat” \( f \)
- \( Q^* \) is not necessarily flat!
- 0 for deterministic transitions. Issue is only serious when env highly stochastic

Workaround #1
Bellman error minimization

\[
E_{(s,a) \sim \nu} \left[ (f(s,a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right))^2 \right]
\]

\[
= E_{(s,a) \sim \nu} \left[ (f(s,a) - (Tf)(s,a))^2 \right] + E_{(s,a) \sim \nu} \left[ \left( (Tf)(s,a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 \right]
\]

This part is what we want: \|f - Tf\|, with a weighted 2-norm defined w/ \( \nu \)

This part is annoying!

• Prefer “flat” \( f \)
• \( Q^* \) is not necessarily flat!
• 0 for deterministic transitions. Issue is only serious when env highly stochastic

Workaround #1

• For \( (s, a) \sim \nu \), if we can obtain 2 i.i.d. copies of \( (r, s') \) (copy A & B):
Bellman error minimization

$$E_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right]$$

$$= E_{(s,a) \sim \nu} \left[ (f(s,a) - (Tf)(s,a))^2 \right] + E_{(s,a) \sim \nu} \left[ (Tf)(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right]^2$$

This part is what we want:
$$\| f - T f \|$$, with a weighted 2-norm defined w/ \( \nu \)

This part is annoying!
- Prefer “flat” \( f \)
- \( Q^* \) is not necessarily flat!
- 0 for deterministic transitions. Issue is only serious when env highly stochastic

Workaround #1
- For \((s,a) \sim \nu\), if we can obtain 2 i.i.d. copies of \((r,s')\) (copy A & B):
  $$\left( f(s,a) - \left( r_A + \gamma \max_{a' \in A} f(s'_A,a') \right) \right) \left( f(s,a) - \left( r_B + \gamma \max_{a' \in A} f(s'_B,a') \right) \right)$$
Bellman error minimization

\[
E_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right] = E_{(s,a) \sim \nu} \left[ (f(s,a) - (Tf)(s,a))^2 \right] + E_{(s,a) \sim \nu} \left[ \left( (Tf)(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right]
\]

This part is what we want: \( \| f - T f \| \), with a weighted 2-norm defined w/ \( \nu \)

This part is annoying:
• Prefer "flat" \( f \)
• \( Q^* \) is not necessarily flat!
• 0 for deterministic transitions. Issue is only serious when env highly stochastic

Workaround #1

• For \((s, a) \sim \nu\), if we can obtain 2 i.i.d. copies of \((r, s')\) (copy A & B):

\[
\left( f(s,a) - \left( r_A + \gamma \max_{a' \in A} f(s'_A,a') \right) \right) \left( f(s,a) - \left( r_B + \gamma \max_{a' \in A} f(s'_B,a') \right) \right)
\]
Bellman error minimization

\[
\mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right]
\]

\[
= \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - (Tf)(s,a) \right)^2 \right] + \mathbb{E}_{(s,a) \sim \nu} \left[ \left( (Tf)(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right]
\]

This part is what we want: \( \| f - T f \| \), with a weighted 2-norm defined w/ \( \nu \)

Unbiased estimate “double sampling”

Workaround #1

• For \((s, a) \sim \nu\), if we can obtain 2 i.i.d. copies of \((r, s')\) (copy A & B):

\[
\left( f(s,a) - \left( r_A + \gamma \max_{a' \in A} f(s'_A, a') \right) \right) \left( f(s,a) - \left( r_B + \gamma \max_{a' \in A} f(s'_B, a') \right) \right)
\]

• Only doable in simulators w/ resetting functionality…
Bellman error minimization

\[ E_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right] \]

\[ = E_{(s,a) \sim \nu} \left[ (f(s,a) - (\mathcal{T} f)(s,a))^2 \right] + E_{(s,a) \sim \nu} \left[ \left( (\mathcal{T} f)(s,a) - \left( r + \gamma \max_{a' \in A} f(s',a') \right) \right)^2 \right] \]

This part is what we want: \( \| f - \mathcal{T} f \| \), with a weighted 2-norm defined w/ \( \nu \)

This part is annoying!
- Prefer “flat” \( f \)
- \( Q^* \) is not necessarily flat!
- 0 for deterministic transitions. Issue is only serious when env highly stochastic

Workaround #2
- Estimate the 2nd part, and subtract it from LHS
Bellman error minimization

\[
E_{(s,a)\sim\nu} \left[ (f(s,a) - (r + \gamma \max_{a'\in A} f(s', a')))^2 \right]
\]

\[
= E_{(s,a)\sim\nu} \left[ (f(s,a) - (Tf)(s,a))^2 \right] + E_{(s,a)\sim\nu} \left[ ((Tf)(s,a) - (r + \gamma \max_{a'\in A} f(s', a')))^2 \right]
\]

This part is what we want: \( \| f - Tf \| \), with a weighted 2-norm defined w/ \( \nu \)

This part is annoying!
- Prefer “flat” \( f \)
- \( Q^* \) is not necessarily flat!
- 0 for deterministic transitions. Issue is only serious when env highly stochastic

Workaround #2
- Estimate the 2nd part, and subtract it from LHS
- [Antos et al’08; Dai et al’18]

\[
E_{(s,a)\sim\nu} \left[ (f(s,a) - (r + \gamma \max_{a'\in A} f(s', a')))^2 \right] - \inf_{g\in G} E_{(s,a)\sim\nu} \left[ (g(s,a) - (r + \gamma \max_{a'\in A} f(s', a')))^2 \right]
\]
Bellman error minimization

$$\inf_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \left( \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s',a') \right) \right)^2 - \left( g(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s',a') \right) \right)^2 \right] \right)$$
Bellman error minimization

\[
\inf_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s',a') \right) \right)^2 - \left( g(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s',a') \right) \right)^2 \right]
\]

- Fix any \( f \), the first squared error is constant; second square is a regression problem w/ Bayes optimal being \( \mathcal{T} f \)
Bellman error minimization

\[
\inf_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \left( \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s',a') \right) \right)^2 - \left( g(s,a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s',a') \right) \right)^2 \right] \right)
\]

- Fix any \( f \), the first squared error is constant; second square is a regression problem w/ Bayes optimal being \( T f \)
- So, if \( G \) is rich enough to contain \( T f \) for all \( f \), this works!
Bellman error minimization

$$\inf_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \left( \mathbb{E}_{(s, a) \sim \nu} \left[ \left( f(s, a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s', a') \right) \right)^2 - \left( g(s, a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s', a') \right) \right)^2 \right] \right)$$

- Fix any $f$, the first squared error is constant; second square is a regression problem w/ Bayes optimal being $\mathcal{T}f$
- So, if $G$ is rich enough to contain $\mathcal{T}f$ for all $f$, this works!
  - and w/ a consistent optimization objective, unlike FQI
Bellman error minimization

\[
\inf_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \left( \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s, a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right) \right)^2 - \left( g(s, a) - \left( r + \gamma \max_{a' \in A} f(s', a') \right)^2 \right] \right) \]

- Fix any \( f \), the first squared error is constant; second square is a regression problem w/ Bayes optimal being \( Tf \)
- So, if \( \mathcal{G} \) is rich enough to contain \( Tf \) for all \( f \), this works!
  - and w/ a consistent optimization objective, unlike FQI
- If \( \mathcal{G} \) is not rich enough, may under-estimate the Bellman error of some \( f \) (subtracting too much)
Bellman error minimization

\[
\inf_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \left( \mathbb{E}_{(s,a) \sim \nu} \left[ \left( f(s, a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s', a') \right) \right)^2 - \left( g(s, a) - \left( r + \gamma \max_{a' \in \mathcal{A}} f(s', a') \right) \right)^2 \right] \right)
\]

- Fix any \( f \), the first squared error is constant; second square is a regression problem w/ Bayes optimal being \( \mathcal{T}f \)
- So, if \( G \) is rich enough to contain \( \mathcal{T}f \) for all \( f \), this works!
  - and w/ a consistent optimization objective, unlike FQI
- If \( G \) is not rich enough, may under-estimate the Bellman error of some \( f \) (subtracting too much)
- FQI: Use \( G=F \); has poly sample complexity guarantee under this assumption
Bellman error minimization

$\mathcal{I} f \in F$ for all $f \in F$
Bellman error minimization

\[ \mathcal{J} f \in F \text{ for all } f \in F \]

• A kind of “completeness”: \( F \) is closed under \( \mathcal{J} \)
Bellman error minimization

\( \mathcal{J} f \in F \) for all \( f \in F \)

- A kind of “completeness”: \( F \) is closed under \( \mathcal{J} \)
- For finite class \( F \), this implies realizability. Why?
Bellman error minimization

\[ \mathcal{I} f \in F \text{ for all } f \in F \]

- A kind of “completeness”: \( F \) is closed under \( \mathcal{I} \)
- For finite class \( F \), this implies realizability. Why?
- What does it mean for abstractions?
Bellman error minimization

\[ \mathcal{T} f \in F \text{ for all } f \in F \]

- A kind of “completeness”: \( F \) is closed under \( \mathcal{T} \)
- For finite class \( F \), this implies realizability. Why?
- What does it mean for abstractions?
  - For piece-wise constant function spaces, completeness = bisimulation (homework)
A simple example (finite horizon, $\gamma=1$)

reward: $\text{Ber}(0.5)$
A simple example (finite horizon, $\gamma=1$)

Dataset $D = \{(s, r, s')\}$ looks like (action omitted):

{([1, 0, 2], [2, 0, 3], ..., [10, 1, end], ..., [10, 0, end])}

Reward: $\text{Ber}(0.5)$
A simple example (finite horizon, $\gamma=1$)

Dataset $D = \{(s, r, s')\}$ looks like (action omitted):

$\{(1, 0, 2), (2, 0, 3), \ldots, (10, 1, \text{end}), \ldots, (10, 0, \text{end})\}$

FQI

Iter #1: Data: $(10, 1, \text{end}), \ldots, (10, 0, \text{end})$

Reward: $\text{Ber}(0.5)$
A simple example (finite horizon, $\gamma=1$)

Dataset $D = \{(s, r, s')\}$ looks like (action omitted):

$\{(1, 0, 2), (2, 0, 3), \ldots, (10, 1, \text{end}), \ldots, (10, 0, \text{end})\}$

FQI
Iter #1: Data: (10, 1, end), ..., (10, 0, end) ➞ 0.501

reward: Ber(0.5)
A simple example (finite horizon, $\gamma=1$)

- Dataset $D = \{(s, r, s')\}$ looks like (action omitted):
  $\{(1, 0, 2), (2, 0, 3), \ldots, (10, 1, \text{end}), \ldots, (10, 0, \text{end})\}$
A simple example (finite horizon, $\gamma=1$)

Dataset $D = \{(s, r, s')\}$ looks like (action omitted):

$\{(1, 0, 2), (2, 0, 3), \ldots, (10, 1, end), \ldots, (10, 0, end)\}$

FQI

Iter #1: Data: $(10, 1, end), \ldots, (10, 0, end)$ $\Rightarrow$ 0.501

Iter #2: Data: $(9, 0, 10)$ $\Rightarrow$ $(9, 0+0.501)$

reward: Ber(0.5)
A simple example (finite horizon, $\gamma=1$)

$\text{Dataset } D = \{(s, r, s')\}$ looks like (action omitted):

$\{(1, 0, 2), (2, 0, 3), \ldots, (10, 1, \text{end}), \ldots, (10, 0, \text{end})\}$

FQI:

Iter #1:

Data: $(10, 1, \text{end}), \ldots, (10, 0, \text{end}) \
\Rightarrow 0.501$

Iter #2:

Data: $(9, 0, 10) \
\Rightarrow (9, 0+0.501) \
\Rightarrow 0.501 0.501$

reward: $\text{Ber}(0.5)$
A simple example (finite horizon, $\gamma=1$)

Dataset $D = \{(s, r, s')\}$ looks like (action omitted):
\[
\{(1, 0, 2), (2, 0, 3), \ldots, (10, 1, \text{end}), \ldots, (10, 0, \text{end})\}
\]

\begin{itemize}
  \item FQI
  \item Iter #1: Data: $(10, 1, \text{end}), \ldots, (10, 0, \text{end})$ \Rightarrow 0.501
  \item Iter #2: Data: $(9, 0, 10) \Rightarrow (9, 0+0.501) \Rightarrow 0.501 0.501$
  \item \ldots
  \item Iter #10: 0.501 0.501 0.501 0.501 \ldots 0.501 0.501 0.501 0.501
\end{itemize}
How things go wrong (w/ restricted class)

Function class

start 1 2 3 4 \ldots 8 9 10

reward: Ber(0.5)
How things go wrong (w/ restricted class)

Function
class

0.5
0.501
How things go wrong (w/ restricted class)

Function class

\[
\begin{array}{cccccccc}
& & & & & & & \\
& & & & & & & \\
& & & & & & & \\
& & & & & & & \\
start & \Rightarrow & 1 & \Rightarrow & 2 & \Rightarrow & 3 & \Rightarrow & 4 & \Rightarrow & \cdots & \Rightarrow & 8 & \Rightarrow & 9 & \Rightarrow & 10 & \\
& & & & & & & \\
& & & & & & & \\
& & & & & & & \\
& & & & & & & \\
0.5 & \times & 0.5 & \\
0.502 & \times & 0.501
\end{array}
\]

reward: Ber(0.5)
How things go wrong (w/ restricted class)

Function class

\[ \begin{array}{c}
0.5 \\
0.504 \\
0.502 \\
0.501
\end{array} \times \begin{array}{c}
0.5 \\
0.5 \\
0.5
\end{array} \]

reward: \text{Ber}(0.5)
How things go wrong (w/ restricted class)

Start: 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow \cdots \rightarrow 8 \rightarrow 9 \rightarrow 10

Function class:
- 0.5 \times 0.5 \times 0.5 \times \cdots \times 0.5 \times 0.5 \times 0.5
- 1.012 \times 0.756 \times 0.628 \times \cdots \times 0.504 \times 0.502 \times 0.501

reward: Ber(0.5)
How things go wrong (w/ restricted class)

start → 1 → 2 → 3 → 4 → ... → 8 → 9 → 10

Function class

| Data: (10, 1, end), ..., (10, 0, end) | 0.501 |

Iter #1: FQI

Data: (10, 1, end), ..., (10, 0, end) → 0.501
How things go wrong (w/ restricted class)

Function class

\[
\begin{align*}
0.5 & \times 0.5 & \times 0.5 & \times \cdots & \times 0.5 & \times 0.5 & \times 0.5 \\
1.012 & \times 0.756 & \times 0.628 & \times \cdots & \times 0.504 & \times 0.502 & \times 0.501
\end{align*}
\]

FQI

Iter #1: \textbf{Data:} \((10, 1, \text{end}), \ldots, (10, 0, \text{end}) \Rightarrow 0.501

Iter #2: \textbf{Data:} \((9, 0, 10) \Rightarrow (9, 0+0.501) \Rightarrow 0.501

\text{reward: } Ber(0.5)
How things go wrong (w/ restricted class)

```
FQI
Iter #1: Data: (10, 1, end), ..., (10, 0, end) ⇒ 0.501

Iter #2: Data: (9, 0, 10) ⇒ (9, 0 + 0.501) ⇒ 0.502 0.501
```

Function class
0.5 × 0.5 × 0.5 × ... × 0.5 × 0.5 × 0.5
1.012 × 0.756 × 0.628 × ... × 0.504 × 0.502 × 0.501

reward: Ber(0.5)
How things go wrong (w/ restricted class)

Function class
0.5 \times 0.5 \times 0.5 \times \ldots \times 0.5 \times 0.5 \times 0.5

FQI
Data: (10, 1, end), \ldots, (10, 0, end) \Rightarrow 0.501

Iter #2:
Data: (9, 0, 10) \Rightarrow (9, 0 + 0.501) \Rightarrow 0.502, 0.501

Iter #10: 1.012, 0.756, 0.628, \ldots, 0.502, 0.501
How things go wrong (w/ restricted class)

start

\[ \begin{array}{cccccccc}
1 & \rightarrow & 2 & \rightarrow & 3 & \rightarrow & 4 & \rightarrow & \cdots & 8 & \rightarrow & 9 & \rightarrow & 10 \\
\end{array} \]

Function

\[
\begin{array}{cccccccc}
0.5 & \times & 0.5 & \times & 0.5 & \times & \cdots & 0.5 & \times & 0.5 & \times & 0.5 \\
1.012 & \times & 0.756 & \times & 0.628 & \times & \cdots & 0.504 & \times & 0.502 & \times & 0.501 \\
\end{array}
\]

FQI

Iter #1:

Data: (10, 1, end), \ldots, (10, 0, end) \Rightarrow 0.501

Iter #2:

Data: (9, 0, 10) \Rightarrow (9, 0 + 0.501) \Rightarrow 0.502 0.501

\ldots

Iter #10: 1.012 0.756 0.628 \cdots 0.502 0.501
How things go wrong (w/ restricted class)

Iter #1:  
Data: $(10, 1, \text{end})$, $\ldots$, $(10, 0, \text{end})$  
FQI class: $0.5 \times 0.5 \times 0.5 \times \ldots$  
Realizable: $0.501$

Iter #2:  
Data: $(9, 0, 10)$  
FQI class: $0.5 \times 0.5 \times 0.5 \times \ldots$  
Realizable: $0.502$  

Iter #10: $1.012 \ 0.756 \ 0.628 \ \ldots$  
Realizable: $0.502$  

$\text{reward: Ber}(0.5)$
How things go wrong (w/ restricted class)

start

1 → 2 → 3 → 4 → … → 8 → 9 → 10

Function class

0.5 0.5 0.5 … 0.5 0.5 0.5

0.501 0.502 0.504 × × × × × × × ×

FQI

Iter #1: Data: (10, 1, end), …, (10, 0, end) ⇒ 0.501

Iter #2: Data: (9, 0, 10) ⇒ (9, 0 + 0.501) ⇒ 0.502 0.501

…

Iter #10: 1.012 0.756 0.628 … 0.502 0.501

Example given in Dann et al’18

• similar conclusion known for decades (e.g., Van Roy’94)
How completeness fixes the issue

Function class

\[
\begin{array}{cc}
0.5 & \times & 0.5 \\
0.756 & \times & 0.628
\end{array}
\]

reward: $\text{Ber}(0.5)$
How completeness fixes the issue

Function class

\[
\begin{array}{c}
0.5 \\
0.628 \\
0.756
\end{array} \times \begin{array}{c}
0.5 \\
0.628
\end{array}
\]

\[
\text{reward: } \text{Ber}(0.5)
\]
How completeness fixes the issue

Reward: Ber(0.5)

Function class

<table>
<thead>
<tr>
<th>0.5</th>
<th>×</th>
<th>0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.628</td>
<td>×</td>
<td>0.628</td>
</tr>
<tr>
<td>0.756</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*introduced by Szepesvari & Munos [2005]

More generally: issue goes away if the regression problem

\[
\left\{ (s, a), (r + \gamma \max_{a' \in A} f_{t-1}(s', a')) \right\}
\]

is realizable with \( F \), for any \( f_{t-1} \in F \)
How completeness fixes the issue

\[
\begin{align*}
\text{Function class} & \quad 0.5 \times 0.5 \\
& \quad 0.628 \times 0.628 \\
& \quad 0.756
\end{align*}
\]

• More generally: issue goes away if the regression problem

\[
\left\{(s, a), (r + \gamma \max_{a' \in A} f_{t-1}(s', a'))\right\}
\]

is realizable with \( F \), for any \( f_{t-1} \in F \)

• In finite-horizon setting: the richer function class you use at a lower level, the more difficult to satisfy realizability at higher level

\[\text{reward: Ber}(0.5)\]

*introduced by Szepesvari & Munos [2005]
How completeness fixes the issue

Function class

• More generally: issue goes away if the regression problem

\[ \left\{ (s, a), \left( r + \gamma \max_{a' \in A} f_{t-1}(s', a') \right) \right\} \]

is realizable with \( F \), for any \( f_{t-1} \in F \)

• In finite-horizon setting: the richer function class you use at a lower level, the more difficult to satisfy realizability at higher level

• In discounted setting: \( F \) closed under Bellman update—adding functions can hurt representation

\[ \text{reward: Ber(0.5)} \]
How completeness fixes the issue

More generally: issue goes away if the regression problem

\[
\left\{ (s, a), (r + \gamma \max_{a' \in A} f_{t-1}(s', a')) \right\}
\]

is realizable with \( F \), for any \( f_{t-1} \in F \)

In finite-horizon setting: the richer function class you use at a lower level, the more difficult to satisfy realizability at higher level

In discounted setting: \( F \) closed under Bellman update—adding functions can hurt representation

With this assumption, sample complexity \( \text{poly}(\log |F|, H, 1/\varepsilon, 1/\delta, C) \)

*introduced by Szepesvari & Munos [2005]
One last assumption: data

• Recall that data needs to be exploratory for batch RL
One last assumption: data

• Recall that data needs to be exploratory for batch RL
• What does it actually mean?
One last assumption: data

- Recall that data needs to be exploratory for batch RL
- What does it actually mean?
  - tabular: relatively uniform over state space
One last assumption: data

- Recall that data needs to be exploratory for batch RL
- What does it actually mean?
  - tabular: relatively uniform over state space
  - abstraction: relatively uniform over abstract state space
One last assumption: data

• Recall that data needs to be exploratory for batch RL
• What does it actually mean?
  • tabular: relatively uniform over state space
  • abstraction: relatively uniform over abstract state space
  • large/continuous state space: uniform? in what measure??
One last assumption: data

• Recall that data needs to be exploratory for batch RL
• What does it actually mean?
  • tabular: relatively uniform over state space
  • abstraction: relatively uniform over abstract state space
  • large/continuous state space: uniform? in what measure??
• Bad news: in the worst case, no distribution would work
One last assumption: data

- Recall that data needs to be exploratory for batch RL
- What does it actually mean?
  - tabular: relatively uniform over state space
  - abstraction: relatively uniform over abstract state space
  - large/continuous state space: uniform? in what measure??
- Bad news: in the worst case, no distribution would work
- Formally:

There exists a family of large MDPs, $F$ that is always realizable, $G$ that includes $Tf$ for any $f \in F$, s.t.: for any data distribution and any batch RL algorithm with $F, G$ as input, the worst-case sample complexity cannot be $\text{poly}(|A|, H, \log |F|, \log |G|)$. 
Proof of lower bound

• Idea: we are allowed unbounded # of states — use a depth-$H$ complete tree to essentially emulate MAB w/ $|A|^H$ arms
Proof of lower bound

• Idea: we are allowed unbounded # of states — use a depth-$H$ complete tree to essentially emulate MAB w/ $|A|^H$ arms

• Recall that sample complexity lower bound for MAB is $\#\text{arms}/\varepsilon^2$
Proof of lower bound

• Idea: we are allowed unbounded # of states — use a depth-$H$ complete tree to essentially emulate MAB w/ $|A|^H$ arms

• Recall that sample complexity lower bound for MAB is $\#\text{arms}/\varepsilon^2$

• Without function approximation: exponential sample complexity for exploration algorithms
Proof of lower bound

• Idea: we are allowed unbounded \# of states — use a depth-$H$ complete tree to essentially emulate MAB w/ $|A|^H$ arms
• Recall that sample complexity lower bound for MAB is $\frac{\#\text{arms}}{\epsilon^2}$
• Without function approximation: exponential sample complexity for exploration algorithms
• Remain to show: (1) function approx. does not help. (2) transfer the claim to batch algorithms
Proof of lower bound

Show: func. approx. does not help:
Proof of lower bound

Show: func. approx. does not help:

- Let $F$ be the collection of $Q^*$ from all MDPs in family

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Show: func. approx. does not help:

• Let $F$ be the collection of $Q^*$ from all MDPs in family
• $\log|F| = H \log|A|$, always realizable

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Show: func. approx. does not help:

- Let $F$ be the collection of $Q^*$ from all MDPs in family
- $\log |F| = H \log |A|$, always realizable
- Let $G$ be the collection of applying $\mathcal{F}$ in every MDP to the $Q^*$ in every other MDP

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Show: func. approx. does not help:

• Let $F$ be the collection of $Q^*$ from all MDPs in family
• $\log|F| = H \log|A|$, always realizable
• Let $G$ be the collection of applying $\mathcal{T}$ in every MDP to the $Q^*$ in every other MDP
• $\log|G| = \log|F|^2 = 2 \log|F|$
Proof of lower bound

Show: func. approx. does not help:

• Let $F$ be the collection of $Q^*$ from all MDPs in family
• $\log|F| = H \log|A|$, always realizable
• Let $G$ be the collection of applying $T$ in every MDP to the $Q^*$ in every other MDP
• $\log|G| = \log|F|^2 = 2 \log|F|$
• In lower bound proof, alg is allowed to specialize to the problem family — giving $F$ and $G$ does not help

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Exploration to batch

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Exploration to batch
• Transition is deterministic (only leaf reward is stochastic)

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Exploration to batch

• Transition is deterministic (only leaf reward is stochastic)
• An exploration algorithm can obtain any data distribution as it likes

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Exploration to batch

- Transition is deterministic (only leaf reward is stochastic)
- An exploration algorithm can obtain any data distribution as it likes
- Any fixed data distribution + batch algorithm = special case of exploration algorithm!

\[ F = \left\{ f(\cdot; \theta) : \theta \in \Theta \right\} \]

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Exploration to batch

• Transition is deterministic (only leaf reward is stochastic)
• An exploration algorithm can obtain any data distribution as it likes
• Any fixed data distribution + batch algorithm = special case of exploration algorithm!
• Lower bound for a set of algorithms automatically applies to its subset — done.

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Proof of lower bound

Summarize:

\[ F = \{ f(\cdot; \mathbf{x}) : \mathbf{x} \in \mathbb{R}^d \} \]

Construction from [Krishnamurthy et al’16];
also used in [Jiang et al’17]
Proof of lower bound

Summarize:

• Sample complexity is $\Omega(|A|^H/\varepsilon^2)$ (because $F$ and $G$ are useless)
Proof of lower bound

Summarize:

- Sample complexity is $\Omega(|A|^H/\varepsilon^2)$ (because $F$ and $G$ are useless)
- $|A|$, $H$, $\log|F|$ and $\log|G|$ are all poly in $|A|$ and $H$

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
Summarize:

- Sample complexity is $\Omega(|A|^H/\varepsilon^2)$ (because $F$ and $G$ are useless)
- $|A|$, $H$, $\log|F|$ and $\log|G|$ are all poly in $|A|$ and $H$
- $\text{poly}(|A|, H, \log|F|, \log|G|)$ is impossible

Construction from [Krishnamurthy et al’16]; also used in [Jiang et al’17]
What’s causing this: Distribution shift

• All paths are symmetric — training data should be uniform (ideally)
What’s causing this: Distribution shift

- All paths are symmetric — training data should be uniform (ideally)
- We incur error “along” the computed policy, which is 1 path
What’s causing this: Distribution shift

• All paths are symmetric — training data should be uniform (ideally)
• We incur error “along” the computed policy, which is 1 path
• States that matter in “test phase” are only given exponentially small weights in training
What’s causing this: Distribution shift

- All paths are symmetric — training data should be uniform (ideally)
- We incur error “along” the computed policy, which is 1 path
- States that matter in “test phase” are only given exponentially small weights in training
- Need assumption on the environment: distribution shift is not severe
Assumption on data: “Concentratability”

\[ d(s, a) / \mu(s, a) \] is upper bounded by a constant \( C \) ("concentratability")

\( \mu \): data distribution

Distribution induced by any policy \( \pi \)

Figure 2: Distribution Shift
Assumption on data: “Concentratability”

- Let $C$ be a uniform upper bound on the density ratio
Assumption on data: “Concentratability”

- Let $C$ be a uniform upper bound on the density ratio.
- Assumption: $C$ is small (allow polynomial dependence on $C$).

*introduced by Munos [2003]
Assumption on data: “Concentratability”

*introduced by Munos [2003]

• Let $C$ be a uniform upper bound on the density ratio
• Assumption: $C$ is small (= allow polynomial dependence on $C$)
• Previous exponential lower bound is “explained away” by an exponentially large $C
Concentratability: when is it small?

Connections to the assumptions needed for efficient exploration [Jiang et al’17]

Markovian high-dimensional observation
Remainder of this part

Prove the $\text{poly}(|A|, H, \log|F|, C)$ result for FQI!
• \( \mathcal{F} f \in F \) for all \( f \in F \) (implies \( Q^* \in F \) for finite class \( F \))
• $\mathcal{I} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)
• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)
• $\mathcal{F}f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)

• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)

• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_{s \in S} \eta_h^\pi(s)/\mu(s) \leq C$, where $\eta_h^\pi(s) := \Pr[s_{h} = s \mid s_1 \sim \rho_0, \pi]$
• $\mathcal{F}f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)

• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim \mathbb{U}$ (Uniform)

• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_{s \in S} \eta^h \pi(s)/\mu(s) \leq C$, where $\eta^h \pi(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$

• Algorithm: $f_{k+1} \leftarrow \arg \min_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where
\[
\mathcal{L}_D(f; f') := \frac{1}{|D|} \sum_{(s, a, r, s') \in D} (f(s, a) - r - \gamma V_{f'}(s'))^2.
\] Also let: $\mathcal{L}_{\mu \times \mathbb{U}}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$. 


• $\mathcal{F} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)
• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)
• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_{s \in S} \eta_h^\pi(s)/\mu(s) \leq C$, where $\eta_h^\pi(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$
• Algorithm: $f_{k+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where
  $$\mathcal{L}_D(f; f') := \frac{1}{|D|} \sum_{(s,a,r,s') \in D} (f(s,a) - r - \gamma V_{f'}(s'))^2 .$$
  Also let: $\mathcal{L}_{\mu \times U}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$.
• For any function $g$ and distribution $\nu$, define $\|g\|_{p,\nu} := (\mathbb{E}_{s \sim \nu}[|g(s)|^p])^{1/p}$
• $\mathcal{F}f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)
• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)
• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_{s} \eta_{h}^{\pi}(s)/\mu(s) \leq C$, where $\eta_{h}^{\pi}(s) := \Pr[s_{h} = s \mid s_{1} \sim \rho_{0}, \pi]$
• Algorithm: $f_{k+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_{D}(f; f_{k})$, where
  $\mathcal{L}_{D}(f; f') := \frac{1}{|D|} \sum_{(s, a, r, s') \in D} (f(s, a) - r - \gamma V_{f'}(s'))^{2}$. Also let: $\mathcal{L}_{\mu \times U}(f; f') := \mathbb{E}_{D}[\mathcal{L}_{D}(f; f')]$.
• For any function $g$ and distribution $\nu$, define $\|g\|_{p, \nu} := (\mathbb{E}_{s \sim \nu}[|g(s)|^{p}])^{1/p}$
  • Monotone in $p$ (“dim normalized”). $p = 2$ if omitted.
• $\mathcal{F} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)

• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)

• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_s \eta_h^{\pi}(s)/\mu(s) \leq C$, where $\eta_h^{\pi}(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$

• Algorithm: $f_{k+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where
  $$\mathcal{L}_D(f; f') := \frac{1}{|D|} \sum_{(s, a, r, s') \in D} (f(s, a) - r - \gamma V_f(s'))^2 \cdot \mathcal{L}_{\mu \times U}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')] \cdot \mathbb{E}_{s \sim \nu}[|g(s)|^p]^{1/p}$$. Also let: $\mathcal{L}_{\mu \times U}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$.

• For any function $g$ and distribution $\nu$, define $\|g\|_{p, \nu} := (\mathbb{E}_{s \sim \nu}[|g(s)|^p])^{1/p}$
  • Monotone in $p$ ("dim normalized"). $p = 2$ if omitted.
  • If $\max_s \nu(s)/\mu(s) \leq C$, $\| \cdot \|_\nu \leq \sqrt{C} \| \cdot \|_\mu$
• $\mathcal{F} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)

• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)

• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_{s \in S} \eta_h^\pi(s)/\mu(s) \leq C$, where $\eta_h^\pi(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$

• Algorithm: $f_{k+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where

$$
\mathcal{L}_D(f; f') := \frac{1}{|D|} \sum_{(s, a, r, s') \in D} (f(s, a) - r - \gamma V_f'(s'))^2.
$$

Also let: $\mathcal{L}_{\mu \times U}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$.

• For any function $g$ and distribution $\nu$, define $\|g\|_{p, \nu} := (\mathbb{E}_{s \sim \nu}[|g(s)|^p])^{1/p}$

  • Monotone in $p$ (“dim normalized”). $p = 2$ if omitted.

  • If $\max_{s \in S} \nu(s)/\mu(s) \leq C$, $\| \cdot \|_{\nu} \leq \sqrt{C}\| \cdot \|_{\mu}$

• Uniform deviation bound: $\forall f, f' \in \mathcal{F}$, $|\mathcal{L}_D(f; f') - \mathcal{L}_{\mu \times U}(f; f')| \leq \epsilon$. 

Notes on Fitted Q-iteration

1. Let $\mathcal{F}$ be a shorthand for $\mathcal{F}_0 \times \mathcal{A} \times \mathcal{D}$.

2. Dataset:

$$
\mathcal{D} := \{(s, a, r, s')\}.
$$

3. Realizability:

Let $\mathcal{F}$ be a dataset where $\mathcal{F} := \{(s, a, r, s')\}$.

4. Let $\mathcal{D}$ be a distribution over states.

5. Let $\mathcal{F}$ be a distribution over states.

6. Let $\mathcal{F}$ be a distribution over states.

7. Let $\mathcal{F}$ be a distribution over states.

8. Let $\mathcal{F}$ be a distribution over states.

9. Let $\mathcal{F}$ be a distribution over states.

10. Let $\mathcal{F}$ be a distribution over states.

11. Let $\mathcal{F}$ be a distribution over states.

12. Let $\mathcal{F}$ be a distribution over states.

13. Let $\mathcal{F}$ be a distribution over states.

14. Let $\mathcal{F}$ be a distribution over states.

15. Let $\mathcal{F}$ be a distribution over states.

16. Let $\mathcal{F}$ be a distribution over states.

17. Let $\mathcal{F}$ be a distribution over states.

18. Let $\mathcal{F}$ be a distribution over states.

19. Let $\mathcal{F}$ be a distribution over states.

20. Let $\mathcal{F}$ be a distribution over states.

21. Let $\mathcal{F}$ be a distribution over states.

22. Let $\mathcal{F}$ be a distribution over states.

23. Let $\mathcal{F}$ be a distribution over states.

24. Let $\mathcal{F}$ be a distribution over states.

25. Let $\mathcal{F}$ be a distribution over states.

26. Let $\mathcal{F}$ be a distribution over states.

27. Let $\mathcal{F}$ be a distribution over states.

28. Let $\mathcal{F}$ be a distribution over states.

29. Let $\mathcal{F}$ be a distribution over states.

30. Let $\mathcal{F}$ be a distribution over states.

31. Let $\mathcal{F}$ be a distribution over states.

32. Let $\mathcal{F}$ be a distribution over states.

33. Let $\mathcal{F}$ be a distribution over states.

34. Let $\mathcal{F}$ be a distribution over states.

35. Let $\mathcal{F}$ be a distribution over states.

36. Let $\mathcal{F}$ be a distribution over states.

37. Let $\mathcal{F}$ be a distribution over states.

38. Let $\mathcal{F}$ be a distribution over states.

39. Let $\mathcal{F}$ be a distribution over states.

40. Let $\mathcal{F}$ be a distribution over states.

41. Let $\mathcal{F}$ be a distribution over states.

42. Let $\mathcal{F}$ be a distribution over states.

43. Let $\mathcal{F}$ be a distribution over states.

44. Let $\mathcal{F}$ be a distribution over states.

45. Let $\mathcal{F}$ be a distribution over states.

46. Let $\mathcal{F}$ be a distribution over states.

47. Let $\mathcal{F}$ be a distribution over states.

48. Let $\mathcal{F}$ be a distribution over states.

49. Let $\mathcal{F}$ be a distribution over states.

50. Let $\mathcal{F}$ be a distribution over states.
- $\mathcal{F} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)
- Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)
- Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_s \eta_h^\pi(s)/\mu(s) \leq C$, where $\eta_h^\pi(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$
- Algorithm: $f_{k+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where
  $\mathcal{L}_D(f; f') := \frac{1}{|D|} \sum_{(s,a,r,s') \in D} (f(s,a) - r - \gamma V_{f'}(s'))^2$. Also let: $\mathcal{L}_{\mu \times U}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$.
- For any function $g$ and distribution $\nu$, define $\|g\|_{p, \nu} := (\mathbb{E}_{s \sim \nu}[|g(s)|^p])^{1/p}$
  - Monotone in $p$ ("dim normalized"). $p = 2$ if omitted.
  - If $\max_s \nu(s)/\mu(s) \leq C$, $\| \cdot \|_\nu \leq \sqrt{C} \| \cdot \|_\mu$
- Uniform deviation bound: $\forall f, f' \in \mathcal{F}$, $|\mathcal{L}_D(f; f') - \mathcal{L}_{\mu \times U}(f; f')| \leq \epsilon$.
  - Can be obtained by Hoeffding’s
• $\mathcal{F} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)

• Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim \mathcal{U}$ (Uniform)

• Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_{s \in S} \eta_h^\pi(s)/\mu(s) \leq C$, where $\eta_h^\pi(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$

• Algorithm: $f_{k+1} \leftarrow \arg\min_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where

$$
\mathcal{L}_D(f; f') := \frac{1}{|D|} \sum_{(s, a, r, s') \in D} (f(s, a) - r - \gamma V_f(s'))^2.
$$

Also let: $\mathcal{L}_\mu \times \mathcal{U}(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$.

• For any function $g$ and distribution $\nu$, define $\|g\|_{p, \nu} := (\mathbb{E}_{s \sim \nu}[|g(s)|^p])^{1/p}$

  • Monotone in $p$ ("dim normalized"). $p = 2$ if omitted.
  • If $\max_{s \in S} \nu(s)/\mu(s) \leq C$, $\| \cdot \|_{\nu} \leq \sqrt{C} \cdot \| \mu$

• Uniform deviation bound: $\forall f, f' \in \mathcal{F}$, $|\mathcal{L}_D(f; f') - \mathcal{L}_\mu \times \mathcal{U}(f; f')| \leq \epsilon$.

  • Can be obtained by Hoeffding’s
  • Improved analysis with fast rates (see note)
\begin{itemize}
\item $\mathcal{F} f \in F$ for all $f \in F$ (implies $Q^* \in F$ for finite class $F$)
\item Dataset: $D = \{(s, a, r, s')\}$, where $s \sim \mu$, $a \sim U$ (Uniform)
\item Let $C$ be a constant s.t. for any (non-stationary) $\pi$ and time step $h \geq 0$, $\max_s \mathbb{E}[\eta_h \pi(s) / \mu(s)] \leq C$, where $\eta_h \pi(s) := \Pr[s_h = s \mid s_1 \sim \rho_0, \pi]$
\item Algorithm: $f_{k+1} \leftarrow \text{arg min}_{f \in \mathcal{F}} \mathcal{L}_D(f; f_k)$, where
\[
\mathcal{L}_D(f; f') = \frac{1}{|D|} \sum_{(s,a,r,s') \in D} (f(s,a) - r - \gamma V_{f'}(s'))^2.
\] Also let: $\mathcal{L}_\mu \times U(f; f') := \mathbb{E}_D[\mathcal{L}_D(f; f')]$.
\item For any function $g$ and distribution $\nu$, define $\|g\|_{p,\nu} := \left(\mathbb{E}_{s \sim \nu}[|g(s)|^p]\right)^{1/p}$
\begin{itemize}
\item Monotone in $p$ (“dim normalized”). $p = 2$ if omitted.
\item If $\max_s \mathbb{E} [\nu(s)/\mu(s)] \leq C$, $\|\cdot\|_{\nu} \leq \sqrt{C} \|\cdot\|_{\mu}$
\end{itemize}
\item Uniform deviation bound: $\forall f, f' \in \mathcal{F}$, $|\mathcal{L}_D(f; f') - \mathcal{L}_\mu \times U(f; f')| \leq \epsilon$.
\begin{itemize}
\item Can be obtained by Hoeffding’s
\item Improved analysis with fast rates (see note)
\end{itemize}
\end{itemize}

**Lemma 1.** Define $\pi_{f, f_k}(s) := \text{arg max}_{a \in A} \max \{f(s, a), f_k(s, a)\}$. Then we have $\forall \nu \in \Delta(S)$,
\[
\|V_f - V_{f_k}\|_{\nu} \leq \|f - f_k\|_{\nu \times \pi_{f, f_k}}.
\]
\[ v^* - v^{\tilde{\pi}} = \sum_{h=1}^{\infty} \gamma^{h-1} \mathbb{E}_{s \sim \eta_h^\pi} [V^*(s) - Q^*(s, \tilde{\pi})] \]

\[ \leq \sum_{h=1}^{\infty} \gamma^{h-1} \mathbb{E}_{s \sim \eta_h^\pi} [Q^*(s, \pi^*) - f_k(s, \pi^*) + f_k(s, \tilde{\pi}) - Q^*(s, \tilde{\pi})] \]

\[ \leq \sum_{h=1}^{\infty} \gamma^{h-1} \left( \| Q^* - f_k \|_{1, \eta_h^\pi \times \pi^*} + \| Q^* - f_k \|_{1, \eta_h^\pi \times \tilde{\pi}} \right) \]

\[ \leq \sum_{h=1}^{\infty} \gamma^{h-1} \left( \| Q^* - f_k \|_{\eta_h^\pi \times \pi^*} + \| Q^* - f_k \|_{\eta_h^\pi \times \tilde{\pi}} \right). \]

(continue on board)