ECE276B: Planning & Learning in Robotics
Lecture 12: Model-Free Control

Nikolay Atanasov

natanasov@ucsd.edu

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering

Outline

Model-Free Policy Iteration

Monte Carlo Policy Iteration

Temporal Difference Policy Iteration

Batch Q-Value Iteration

Model-Free Generalized Policy Iteration

- **Model-based case**: Our main tool for stochastic infinite-horizon problems over MDPs with known models is Generalized Policy Iteration (GPI):
  - **Policy Evaluation**: Given π, compute V^π:

$$V^\pi(\mathbf{x}) = \ell(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}_{\mathbf{x}' \sim p_f(\cdot|\mathbf{x},\pi(\mathbf{x}))}\left[V^\pi(\mathbf{x}')\right], \quad \forall \mathbf{x} \in \mathcal{X}$$

  - **Policy Improvement**: Given V^π obtain a new policy π':

$$\pi'(\mathbf{x}) \in \arg\min_{\mathbf{u}\in\mathcal{U}(\mathbf{x})} \underbrace{\left\{\ell(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}_{\mathbf{x}' \sim p_f(\cdot|\mathbf{x},\mathbf{u})}\left[V^\pi(\mathbf{x}')\right]\right\}}_{Q^\pi(\mathbf{x},\mathbf{u})}, \quad \forall \mathbf{x} \in \mathcal{X}$$

- **Model-free case**: Is it still possible to implement the GPI algorithm?
  - **Policy Evaluation**: Given π, MC or TD learning from Lecture 11 can be used to estimate V^π or Q^π
  - **Policy Improvement**: Computing π' based on V^π requires access to $\ell(\mathbf{x}, \mathbf{u})$, $p_f(\mathbf{x}', \mathbf{x}, \mathbf{u})$ but based on Q^π can be done <u>without knowing</u> $\ell(\mathbf{x}, \mathbf{u})$, $p_f(\mathbf{x}', \mathbf{x}, \mathbf{u})$:

$$\pi'(\mathbf{x}) \in \arg\min_{\mathbf{u}\in\mathcal{U}(\mathbf{x})} Q^\pi(\mathbf{x}, \mathbf{u})$$

Policy Evaluation (Recap)

- Given π, iterate $\mathcal{B}_\pi$ to compute V^π or Q^π via Dynamic Programming (DP), Temporal Difference (TD), or Monte Carlo (MC)
- DP needs the models $\ell(\mathbf{x}_t, \mathbf{u}_t)$, $p_f(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ while MC and TD are model-free and use samples $\mathbf{x}_t, \mathbf{u}_t, \ell_t, \mathbf{x}_{t+1}$ instead
- V^π **Policy Evaluation**:

$$DP : \mathcal{B}_\pi[V](\mathbf{x}_t) = \ell(\mathbf{x}_t, \pi(\mathbf{x}_t)) + \gamma \mathbb{E}_{\mathbf{x}_{t+1} \sim p_f(\cdot|\mathbf{x}_t, \pi(\mathbf{x}_t))}\left[V(\mathbf{x}_{t+1})\right]$$

$$TD : \mathcal{B}_\pi[V](\mathbf{x}_t) \approx V(\mathbf{x}_t) + \alpha\left[\ell(\mathbf{x}_t, \mathbf{u}_t) + \gamma V(\mathbf{x}_{t+1}) - V(\mathbf{x}_t)\right]$$

$$MC : \mathcal{B}_\pi[V](\mathbf{x}_t) \approx V(\mathbf{x}_t) + \alpha\left[\sum_{k=0}^{T-t-1} \gamma^k \ell(\mathbf{x}_{t+k}, \mathbf{u}_{t+k}) + \gamma^{T-t}\mathfrak{q}(\mathbf{x}_T) - V(\mathbf{x}_t)\right]$$

- Q^π **Policy Evaluation**:

$$DP : \mathcal{B}_\pi[Q](\mathbf{x}_t, \mathbf{u}_t) = \ell(\mathbf{x}_t, \mathbf{u}_t) + \gamma \mathbb{E}_{\mathbf{x}_{t+1} \sim p_f(\cdot|\mathbf{x}_t, \mathbf{u}_t)}\left[Q(\mathbf{x}_{t+1}, \pi(\mathbf{x}_{t+1}))\right]$$

$$TD : \mathcal{B}_\pi[Q](\mathbf{x}_t, \mathbf{u}_t) \approx Q(\mathbf{x}_t, \mathbf{u}_t) + \alpha\left[\ell(\mathbf{x}_t, \mathbf{u}_t) + \gamma Q(\mathbf{x}_{t+1}, \mathbf{u}_{t+1}) - Q(\mathbf{x}_t, \mathbf{u}_t)\right]$$

$$MC : \mathcal{B}_\pi[Q](\mathbf{x}_t, \mathbf{u}_t) \approx Q(\mathbf{x}_t, \mathbf{u}_t) + \alpha\left[\sum_{k=0}^{T-t-1} \gamma^k \ell(\mathbf{x}_{t+k}, \mathbf{u}_{t+k}) + \gamma^{T-t}\mathfrak{q}(\mathbf{x}_T) - Q(\mathbf{x}_t, \mathbf{u}_t)\right]$$

Model-Free Policy Improvement

- If Q^π, instead of V^π, is estimated via MC or TD, then the policy improvement step can be implemented model-free, i.e., can compute $\min_{\mathbf{u}} Q^\pi(\mathbf{x}, \mathbf{u})$ without knowing the motion model p_f or the stage cost ℓ
- Since $Q^\pi(\mathbf{x}, \mathbf{u})$ computed by MC or TD is an approximation to the true Q function, we might not get an improved policy with respect to the true Q function:
  - picking the "best" control according to the current estimate of Q^π might not be the actual best control
  - if a deterministic policy $\pi(\mathbf{x})$ is used for Evaluation and Improvement, we will observe returns for only one of the possible controls at each state and might not visit many states; estimating Q^π will not be possible at those never-visited states and controls

Example

- There are two doors in front of you
- You open the left door and get reward 0
  $\ell(\mathit{left}) = 0$
- You open the right door and get reward $+1$
  $\ell(\mathit{right}) = -1$
- You open the right door and get reward $+3$
  $\ell(\mathit{right}) = -3$
- You open the right door and get reward $+2$
  $\ell(\mathit{right}) = -2$
- Which door is the best long-term choice?

"Behind one door is tenure - behind the other is flipping burgers at McDonald's."

Model-Free Control

- Two ideas to ensure that we do not commit to wrong controls due to approximation error in Q^{π} too early and continue exploring the state and control space:

    1. **Exploring Starts**: in each episode $\rho^{(k)} \sim \pi$, choose initial state-control pairs randomly with non-zero probability among all possible pairs in $\mathcal{X} \times \mathcal{U}$

    2. ϵ-**Soft Policy**: a **stochastic policy** $\pi(\mathbf{u}|\mathbf{x})$ under which every control has a non-zero probability of being chosen and hence every reachable state will have non-zero probability of being encountered

- **Deterministic Stationary Policy**: function $\pi : \mathcal{X} \rightarrow \mathcal{U}$

- **Stochastic Stationary Policy**: function $\pi : \mathcal{X} \rightarrow \mathcal{P}(\mathcal{U})$, where $\mathcal{P}(\mathcal{U})$ is the set of probability density functions on $\mathcal{U}$:

$$\pi(\mathbf{u}|\mathbf{x}) \geq 0 \qquad \int_{\mathcal{U}} \pi(\mathbf{u}|\mathbf{x}) d\mathbf{u} = 1$$

Outline

First-Visit MC Policy Iteration with Exploring Starts

Algorithm MC Policy Iteration with Exploring Starts

Initialize: $Q(\mathbf{x}, \mathbf{u}), \pi(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$ and $\mathbf{u} \in \mathcal{U}$
loop
    Choose $(\mathbf{x}_0, \mathbf{u}_0) \in \mathcal{X} \times \mathcal{U}$ randomly $\triangleright$ exploring starts
    Generate an episode $\rho = \mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_{T-1}, \mathbf{u}_{t-1}, \mathbf{x}_T$ from π
    for each $\mathbf{x}, \mathbf{u}$ in ρ **do**
        $L \leftarrow$ return following the first occurrence of $\mathbf{x}, \mathbf{u}$
        $Q(\mathbf{x}, \mathbf{u}) \leftarrow Q(\mathbf{x}, \mathbf{u}) + \alpha \left(L - Q(\mathbf{x}, \mathbf{u}) \right)$
    for each $\mathbf{x}$ in ρ **do**
        $\pi(\mathbf{x}) \leftarrow \arg\min_{\mathbf{u}} Q(\mathbf{x}, \mathbf{u})$

ϵ-Greedy Exploration

- An alternative to exploring starts
- To ensure exploration it must be possible to encounter all control $\mathcal{U}$ controls with non-zero probability
- Assume $|\mathcal{U}| < \infty$
- **ϵ-Soft Policy**: stochastic policy that picks each $\mathbf{u}$ with at least $\frac{\epsilon}{|\mathcal{U}|}$ probability:

$$\pi(\mathbf{u}|\mathbf{x}) = \mathbb{P}(\mathbf{u}_t = \mathbf{u} \mid \mathbf{x}_t = \mathbf{x}) \geq \frac{\epsilon}{|\mathcal{U}|} \qquad \forall \mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}$$

- **ϵ-Greedy Policy**: an ϵ-soft policy that picks the best control according to $Q(\mathbf{x}, \mathbf{u})$ in the policy improvement step but ensures that all other controls are selected with at least $\frac{\epsilon}{|\mathcal{U}|}$ probability:

$$\pi(\mathbf{u} \mid \mathbf{x}) = \mathbb{P}(\mathbf{u}_t = \mathbf{u} \mid \mathbf{x}_t = \mathbf{x}) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{U}|} & \text{if } \mathbf{u} = \arg\min_{\mathbf{u}' \in \mathcal{U}} Q(\mathbf{x}, \mathbf{u}') \\ \frac{\epsilon}{|\mathcal{U}|} & \text{otherwise} \end{cases}$$

Bellman Equations with a Stochastic Policy

- **Value function** of a stochastic policy π:

$$V^\pi(\mathbf{x}) := \mathbb{E}_{\mathbf{u}_0,\mathbf{x}_1,\mathbf{u}_1,\mathbf{x}_2,\ldots}\left[\sum_{t=0}^{\infty}\gamma^t \ell(\mathbf{x}_t,\mathbf{u}_t) \mid \mathbf{x}_0 = \mathbf{x}\right]$$
$$= \mathbb{E}_{\mathbf{u}\sim\pi(\cdot|\mathbf{x})}\left[\ell(\mathbf{x},\mathbf{u}) + \gamma\mathbb{E}_{\mathbf{x}'\sim p_f(\cdot|\mathbf{x},\mathbf{u})}\left[V^\pi(\mathbf{x}')\right]\right]$$
$$= \mathbb{E}_{\mathbf{u}\sim\pi(\cdot|\mathbf{x})}\left[Q^\pi(\mathbf{x},\mathbf{u})\right]$$

- **Q function** of a stochastic policy π:

$$Q^\pi(\mathbf{x},\mathbf{u}) := \ell(\mathbf{x},\mathbf{u}) + \mathbb{E}_{\mathbf{x}_1,\mathbf{u}_1,\ldots}\left[\sum_{t=1}^{\infty}\gamma^t \ell(\mathbf{x}_t,\mathbf{u}_t) \mid \mathbf{x}_0 = \mathbf{x}, \mathbf{u}_0 = \mathbf{u}\right]$$
$$= \ell(\mathbf{x},\mathbf{u}) + \gamma\mathbb{E}_{\mathbf{x}'\sim p_f(\cdot|\mathbf{x},\mathbf{u}),\mathbf{u}'\sim\pi(\cdot|\mathbf{x}')}\left[Q^\pi(\mathbf{x}',\mathbf{u}')\right]$$

ϵ-Greedy Policy Improvement

Theorem: ϵ-Greedy Policy Improvement

For any ϵ-soft policy π with associated Q^π, the ϵ-greedy policy π' with respect to Q^π is an improvement, i.e., $V^{\pi'}(\mathbf{x}) \leq V^\pi(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$

- **Proof**:

$$
\begin{aligned}
\mathbb{E}_{\mathbf{u}' \sim \pi'(\cdot|\mathbf{x})}\left[Q^\pi(\mathbf{x}, \mathbf{u}')\right] &= \sum_{\mathbf{u}' \in \mathcal{U}} \pi'(\mathbf{u}' \mid \mathbf{x}) Q^\pi(\mathbf{x}, \mathbf{u}') \\
&= \frac{\epsilon}{|\mathcal{U}|} \sum_{\mathbf{u}' \in \mathcal{U}} Q^\pi(\mathbf{x}, \mathbf{u}') + (1-\epsilon) \min_{\mathbf{u} \in \mathcal{U}} Q^\pi(\mathbf{x}, \mathbf{u}) \\
&\leq \frac{\epsilon}{|\mathcal{U}|} \sum_{\mathbf{u}' \in \mathcal{U}} Q^\pi(\mathbf{x}, \mathbf{u}') + (1-\epsilon) \sum_{\mathbf{u} \in \mathcal{U}} \frac{\pi(\mathbf{u} \mid \mathbf{x}) - \frac{\epsilon}{|\mathcal{U}|}}{1-\epsilon} Q^\pi(\mathbf{x}, \mathbf{u}) \\
&= \sum_{\mathbf{u} \in \mathcal{U}} \pi(\mathbf{u} \mid \mathbf{x}) Q^\pi(\mathbf{x}, \mathbf{u}) = V^\pi(\mathbf{x})
\end{aligned}
$$

ϵ-Greedy Policy Improvement

- Then, similarity to the policy improvement theorem for deterministic policies, for all $\mathbf{x} \in \mathcal{X}$:

$$
\begin{aligned}
V^{\pi}(\mathbf{x}) &\geq \mathbb{E}_{\mathbf{u}_0 \sim \pi'(\cdot|\mathbf{x})}\left[Q^{\pi}(\mathbf{x}, \mathbf{u}_0)\right] \\
&= \mathbb{E}_{\mathbf{u}_0 \sim \pi'(\cdot|\mathbf{x})}\left[\ell(\mathbf{x}, \mathbf{u}_0) + \gamma \mathbb{E}_{\mathbf{x}_1 \sim p_f(\cdot|\mathbf{x}, \mathbf{u}_0)}\left[V^{\pi}(\mathbf{x}_1)\right]\right] \\
&\geq \mathbb{E}_{\mathbf{u}_0 \sim \pi'(\cdot|\mathbf{x})}\left[\ell(\mathbf{x}, \mathbf{u}_0) + \gamma \mathbb{E}_{\mathbf{x}_1 \sim p_f(\cdot|\mathbf{x}, \mathbf{u}_0)}\left[\mathbb{E}_{\mathbf{u}_1 \sim \pi'(\cdot|\mathbf{x}_1)}\left[Q^{\pi}(\mathbf{x}_1, \mathbf{u}_1)\right]\right]\right] \\
&= \mathbb{E}_{\mathbf{u}_0 \sim \pi'(\cdot|\mathbf{x})}\left[\ell(\mathbf{x}, \mathbf{u}_0) + \gamma \mathbb{E}_{\mathbf{x}_1, \mathbf{u}_1}\left[\ell(\mathbf{x}_1, \mathbf{u}_1) + \gamma \mathbb{E}_{\mathbf{x}_2} V^{\pi}(\mathbf{x}_2)\right]\right] \\
&\geq \cdots \geq \mathbb{E}_{\rho_0 \sim \pi'}\left[\sum_{t=0}^{\infty} \gamma^t \ell(\mathbf{x}_t, \mathbf{u}_t) \middle| \mathbf{x}_0 = \mathbf{x}\right] = V^{\pi'}(\mathbf{x})
\end{aligned}
$$

First-Visit MC Policy Iteration with ϵ-Greedy Improvement

Algorithm First-Visit MC Policy Iteration with ϵ-Greedy Improvement

Init: $Q(\mathbf{x}, \mathbf{u})$, $\pi(\mathbf{u}|\mathbf{x})$ (ϵ-soft policy) for all $\mathbf{x} \in \mathcal{X}$ and $\mathbf{u} \in \mathcal{U}$
loop
  Generate an episode $\rho := \mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_{T-1}, \mathbf{u}_{t-1}, \mathbf{x}_T$ from π
  for each $\mathbf{x}, \mathbf{u}$ in ρ **do**
    $L \leftarrow$ return following the first occurrence of $\mathbf{x}, \mathbf{u}$
    $Q(\mathbf{x}, \mathbf{u}) \leftarrow Q(\mathbf{x}, \mathbf{u}) + \alpha \left(L - Q(\mathbf{x}, \mathbf{u})\right)$
  for each $\mathbf{x}$ in ρ **do**
    $\mathcal{U}^* \leftarrow \arg\min_{\mathbf{u}} Q(\mathbf{x}, \mathbf{u})$
    $\pi(\mathbf{u}|\mathbf{x}) \leftarrow \begin{cases} \frac{1-\epsilon}{|\mathcal{U}^*|} + \frac{\epsilon}{|\mathcal{U}(\mathbf{x})|} & \text{if } \mathbf{u} \in \mathcal{U}^* \\ \frac{\epsilon}{|\mathcal{U}(\mathbf{x})|} & \text{if } \mathbf{u} \neq \mathbf{u}^* \end{cases}$

Outline

Temporal-Difference Control

- TD prediction has several advantages over MC prediction:
  - works with incomplete episodes
  - can perform online updates to Q^π after every transition
  - TD estimate of Q^π has lower variance than the MC one
- TD in the policy iteration algorithm:
  - use TD for policy evaluation
  - can update $Q(\mathbf{x}, \mathbf{u})$ after every transition within an episode
  - use an ϵ-greedy policy for policy improvement because we still need to trade off exploration and exploitation

TD Policy Iteration with ϵ-Greedy Improvement (SARSA)

- **SARSA**: estimates Q^{π} using TD updates after every $S_t, A_t, R_t, S_{t+1}, A_{t+1}$ transition:

$$Q(\mathbf{x}_t, \mathbf{u}_t) \leftarrow Q(\mathbf{x}_t, \mathbf{u}_t) + \alpha \left[\ell(\mathbf{x}_t, \mathbf{u}_t) + \gamma Q(\mathbf{x}_{t+1}, \mathbf{u}_{t+1}) - Q(\mathbf{x}_t, \mathbf{u}_t)\right]$$

- Ensures exploration via an ϵ-greedy policy in the policy improvement step

---

Algorithm SARSA

Init: $Q(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}$ and all $\mathbf{u} \in \mathcal{U}$
loop
    $\pi \leftarrow \epsilon$-greedy policy derived from Q
    Generate episode $\rho := \mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_{T-1}, \mathbf{u}_{t-1}, \mathbf{x}_T$ from π
    for $(\mathbf{x}, \mathbf{u}, \mathbf{x}', \mathbf{u}') \in \rho$ **do**
        $Q(\mathbf{x}, \mathbf{u}) \leftarrow Q(\mathbf{x}, \mathbf{u}) + \alpha \left[\ell(\mathbf{x}, \mathbf{u}) + \gamma Q(\mathbf{x}', \mathbf{u}') - Q(\mathbf{x}, \mathbf{u})\right]$

Convergence of Model-Free Policy Iteration

- **Greedy in the Limit with Infinite Exploration** (GLIE):
  - Number of visits to all state-control pairs approach infinity, i.e., all state-control pairs are explored infinitely many times: $\lim_{k\to\infty} N_k(\mathbf{x},\mathbf{u}) = \infty$
  - The ϵ-greedy policy converges to a greedy policy wrt $\mathbf{u}^* \in \arg\min_{\mathbf{u}\in\mathcal{U}(\mathbf{x})} Q(\mathbf{x},\mathbf{u})$
- Example: ϵ-greedy is GLIE with $\epsilon_k = \frac{1}{k}$

$$\pi_k(\mathbf{u} \mid \mathbf{x}) = \begin{cases} 1 - \epsilon_k + \frac{\epsilon_k}{|\mathcal{U}|} & \text{if } \mathbf{u} = \mathbf{u}^* \\ \frac{\epsilon_k}{|\mathcal{U}|} & \text{if } \mathbf{u} \neq \mathbf{u}^* \end{cases} \qquad \lim_{k\to\infty} \pi_k(\mathbf{u} \mid \mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{u} = \mathbf{u}^* \\ 0 & \text{if } \mathbf{u} \neq \mathbf{u}^* \end{cases}$$

Theorem: Convergence of Model-Free Policy Iteration

Both MC Policy Iteration and SARSA converge to the optimal action-value function, $Q(\mathbf{x},\mathbf{u}) \to Q^*(\mathbf{x},\mathbf{u})$, as the number of episodes $k \to \infty$ as long as:

- the sequence of ϵ-greedy policies $\pi_k(\mathbf{u} \mid \mathbf{x})$ is GLIE,
- the sequence of step sizes α_k is Robbins-Monro.

On-Policy vs Off-Policy Learning

- **On-policy prediction**: estimate V^π or Q^π using episodes from π
- **Off-policy prediction**: estimate V^π or Q^π using episodes from μ
- On-policy learning methods:
  - evaluate or improve a policy π that is used to both make decisions and collect experience
  - require well-designed exploration functions
  - empirically successful with function approximation
- Off-policy learning methods:
  - evaluate or improve a policy π that is different from the policy μ used to generate data
  - can use an effective exploration policy μ to generate data while learning an optimal policy π
  - can learn from observing other agents
  - can re-use experience from old policies $\pi_1, \pi_2, \ldots, \pi_{k-1}$
  - can learn about multiple policies while following one policy
  - causes theoretical challenges with function approximation

Importance Sampling for Off-Policy Learning

- Off-policy learning: use episodes generated from μ to evaluate π
- The stage costs obtained from μ need to be re-weighted according to the likelihood that the same states would be encountered by π
- **Importance Sampling**: estimates the expectation of a function $\ell(\mathbf{x})$ with respect to a probability density function $p(\mathbf{x})$ by computing a re-weighted expectation over a different probability density $q(\mathbf{x})$:

$$\begin{aligned}\mathbb{E}_{\mathbf{x}\sim p(\cdot)}[\ell(\mathbf{x})] &= \int p(\mathbf{x})\ell(\mathbf{x})d\mathbf{x} \\ &= \int q(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}\ell(\mathbf{x})d\mathbf{x} = \mathbb{E}_{\mathbf{x}\sim q(\cdot)}\left[\frac{p(\mathbf{x})}{q(\mathbf{x})}\ell(\mathbf{x})\right]\end{aligned}$$

Requires that $q(\mathbf{x}) \neq 0$ when $p(\mathbf{x}) \neq 0$.

Importance Sampling for Off-Policy MC Learning

- To use returns generated from μ to evaluate π via MC, re-weight the long-term cost L_t via importance-sampling corrections along the whole episode:

$$L_t^{\pi/\mu} = \frac{\pi(\mathbf{u}_t|\mathbf{x}_t)}{\mu(\mathbf{u}_t|\mathbf{x}_t)} \frac{\pi(\mathbf{u}_{t+1}|\mathbf{x}_{t+1})}{\mu(\mathbf{u}_{t+1}|\mathbf{x}_{t+1})} \cdots \frac{\pi(\mathbf{u}_{T-1}|\mathbf{x}_{T-1})}{\mu(\mathbf{u}_{T-1}|\mathbf{x}_{T-1})} L_t$$

- This requires that μ should not be zero for any of state-control pairs along the episode from π

- Update the value estimate towards the *corrected long-term cost* $L_t^{\pi/\mu}$:

$$V^\pi(\mathbf{x}_t) \leftarrow V^\pi(\mathbf{x}_t) + \alpha \left(L_t^{\pi/\mu} - V^\pi(\mathbf{x}_t) \right)$$

- **Note**: importance sampling in MC can dramatically increase variance

Importance Sampling for Off-Policy TD Learning

- To use returns generated from μ to evaluate π via TD, re-weight the TD target $\ell(\mathbf{x}, \mathbf{u}) + \gamma V(\mathbf{x}')$ by importance sampling:

$$V^\pi(\mathbf{x}_t) \leftarrow V^\pi(\mathbf{x}_t) + \alpha \left(\frac{\pi(\mathbf{u}_t \mid \mathbf{x}_t)}{\mu(\mathbf{u}_t \mid \mathbf{x}_t)} \left(\ell(\mathbf{x}_t, \mathbf{u}_t) + \gamma V^\pi(\mathbf{x}_{t+1})\right) - V^\pi(\mathbf{x}_t) \right)$$

- Importance sampling in TD is much lower variance than in MC and the policies need to be similar (i.e., μ should not be zero when π is non-zero) over a single step only

Off-Policy TD Control without Importance Sampling

- **Q-Learning** (Watkins, 1989): one of the early breakthroughs in reinforcement learning was the development of an off-policy TD algorithm that does not use importance sampling
- Q-Learning approximates $\mathcal{B}_*[Q](\mathbf{x}, \mathbf{u})$ directly using samples:

$$Q(\mathbf{x}_t, \mathbf{u}_t) \leftarrow Q(\mathbf{x}_t, \mathbf{u}_t) + \alpha \left[\ell(\mathbf{x}_t, \mathbf{u}_t) + \gamma \min_{\mathbf{u} \in \mathcal{U}} Q(\mathbf{x}_{t+1}, \mathbf{u}) - Q(\mathbf{x}_t, \mathbf{u}_t) \right]$$

- The learned Q function approximates Q^* **regardless of the policy being followed**!

Theorem: Convergence of Q-Learning

Q-Learning converges almost surely to Q^* assuming all state-control pairs continue to be updated and the sequence of step sizes α_k is Robbins-Monro.

- C. J. Watkins and P. Dayan. "Q-learning," Machine learning, 1992.

Q-Learning: Off-Policy TD Learning of $Q^*(\mathbf{x}, \mathbf{u})$

Algorithm Q-Learning

Init: $Q(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}$ and all $\mathbf{u} \in \mathcal{U}$
loop
  $\pi \leftarrow \epsilon$-greedy policy derived from Q ▷ π can be arbitrary!
  Generate episode $\rho := \mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \dots, \mathbf{x}_{T-1}, \mathbf{u}_{t-1}, \mathbf{x}_T$ from π
  for $(\mathbf{x}, \mathbf{u}, \mathbf{x}') \in \rho$ **do**
    $Q(\mathbf{x}, \mathbf{u}) \leftarrow Q(\mathbf{x}, \mathbf{u}) + \alpha \left[\ell(\mathbf{x}, \mathbf{u}) + \gamma \min_{\mathbf{u}'} Q(\mathbf{x}', \mathbf{u}') - Q(\mathbf{x}, \mathbf{u})\right]$

Relationship Between Full and Sample Backups

Full Backups (DP)	**Sample Backups** (TD)
Policy Evaluation $V(\mathbf{x}) \leftarrow \mathcal{B}_\pi[V](\mathbf{x}) = \ell(\mathbf{x}, \pi(\mathbf{x})) + \gamma \mathbb{E}_{\mathbf{x}'}[V(\mathbf{x}')]$	**TD Policy Evaluation** $V(\mathbf{x}) \leftarrow V(\mathbf{x}) + \alpha(\ell(\mathbf{x}, \mathbf{u}) + \gamma V(\mathbf{x}') - V(\mathbf{x}))$
Policy Q-Evaluation $Q(\mathbf{x}, \mathbf{u}) \leftarrow \mathcal{B}_\pi[Q](\mathbf{x}, \mathbf{u}) = \ell(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}_{\mathbf{x}'}[Q(\mathbf{x}', \pi(\mathbf{x}'))]$	**TD Policy Q-Evaluation (SARSA)** $Q(\mathbf{x}, \mathbf{u}) \leftarrow Q(\mathbf{x}, \mathbf{u}) + \alpha(\ell(\mathbf{x}, \mathbf{u}) + \gamma Q(\mathbf{x}', \mathbf{u}') - Q(\mathbf{x}, \mathbf{u}))$
Value Iteration $V(\mathbf{x}) \leftarrow \mathcal{B}_*[V](\mathbf{x}) = \min_{\mathbf{u}} \{\ell(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}_{\mathbf{x}'}[V(\mathbf{x}')]\}$	**N/A**
Q-Value Iteration $Q(\mathbf{x}, \mathbf{u}) \leftarrow \mathcal{B}_*[Q](\mathbf{x}, \mathbf{u}) = \ell(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}_{\mathbf{x}'}\left[\min_{\mathbf{u}'} Q(\mathbf{x}', \mathbf{u}')\right]$	**Q-Learning** $Q(\mathbf{x}, \mathbf{u}) \leftarrow Q(\mathbf{x}, \mathbf{u}) + \alpha\left(\ell(\mathbf{x}, \mathbf{u}) + \gamma \min_{\mathbf{u}'} Q(\mathbf{x}', \mathbf{u}') - Q(\mathbf{x}, \mathbf{u})\right)$

Outline

Batch Sampling-Based Q-Value Iteration

Algorithm Batch Sampling-Based Q-Value Iteration

Init: $Q_0(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}$ and all $\mathbf{u} \in \mathcal{U}$
loop
  $\pi \leftarrow \epsilon$-greedy policy derived from Q_i ▷ π can be arbitrary!
  Generate episodes $\{\rho^{(k)}\}_{k=1}^{K}$ from π
  for $(\mathbf{x}, \mathbf{u}) \in \mathcal{X} \times \mathcal{U}$ **do**

$$Q_{i+1}(\mathbf{x}, \mathbf{u}) = \frac{1}{K} \sum_{k=1}^{K} \frac{\sum_{t=0}^{T^{(k)}} \mathcal{B}_*[Q_i](\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}, \mathbf{x}_{t+1}^{(k)}) \mathbb{1}\{(\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}) = (\mathbf{x}, \mathbf{u})\}}{\sum_{t=0}^{T^{(k)}} \mathbb{1}\{(\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}) = (\mathbf{x}, \mathbf{u})\}}$$

- Batch sampling-based Q-value iteration behaves like $Q_{i+1} = \mathcal{B}_*[Q_i] + \text{noise}$. Does it actually converge?

Batch Least-Squares Q-Value Iteration

- $Q_{i+1}(\mathbf{x}, \mathbf{u}) = \textbf{mean}\left\{\mathcal{B}_*[Q_i](\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}, \mathbf{x}_{t+1}^{(k)}),\ \forall k, t \text{ such that } (\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}) = (\mathbf{x}, \mathbf{u})\right\}$
- Note that: $\textbf{mean}\left\{\mathbf{x}^{(k)}\right\} = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \|\mathbf{x}^{(k)} - \mathbf{x}\|^2$
- $Q_{i+1}(\mathbf{x}, \mathbf{u}) = \arg\min_{q} \sum_{k=1}^{K} \sum_{(\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}) = (\mathbf{x}, \mathbf{u})} \left\|\mathcal{B}_*[Q_i](\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}, \mathbf{x}_{t+1}^{(k)}) - q\right\|^2$
- $Q_{i+1}(\cdot, \cdot) = \arg\min_{Q(\cdot,\cdot)} \sum_{k=1}^{K} \sum_{t=0}^{T^{(k)}} \left\|\mathcal{B}_*[Q_i](\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}, \mathbf{x}_{t+1}^{(k)}) - Q(\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)})\right\|^2$

Algorithm Batch Least-Squares Q-Value Iteration

1: **Init**: $Q_0(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}$ and all $\mathbf{u} \in \mathcal{U}$
2: **loop**
3: $\quad \pi \leftarrow \epsilon$-greedy policy derived from Q_i $\quad \triangleright$ π can be arbitrary!
4: $\quad$ Generate episodes $\{\rho^{(k)}\}_{k=1}^{K}$ from π
5: $$Q_{i+1}(\cdot, \cdot) = \arg\min_{Q(\cdot,\cdot)} \sum_{k=1}^{K} \sum_{t=0}^{T^{(k)}} \left\|\mathcal{B}_*[Q_i](\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}, \mathbf{x}_{t+1}^{(k)}) - Q(\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)})\right\|^2$$

Small Steps in the Backup Direction

- **Full backup**: $Q_{i+1} \leftarrow \mathcal{B}_*[Q_i] + \text{noise}$
- **Partial backup**: $Q_{i+1} \leftarrow \alpha\mathcal{B}_*[Q_i] + (1-\alpha)Q_i + \text{noise}$
- Equivalent to a gradient step on a squared error objective function:

$$\begin{aligned} Q_{i+1} &\leftarrow \alpha\mathcal{B}_*[Q_i] + (1-\alpha)Q_i + \text{noise} \\ &= Q_i + \alpha\left(\mathcal{B}_*[Q_i] - Q_i\right) + \text{noise} \\ &= Q_i - \alpha\left(\frac{1}{2}\nabla_Q \|\mathcal{B}_*[Q_i] - Q\|^2\Big|_{Q=Q_i} + \text{noise}\right) \end{aligned}$$

- Behaves like stochastic gradient descent for $f(Q) := \frac{1}{2}\|\mathcal{B}_*[Q_i] - Q\|^2$ but the objective is changing because $\mathcal{B}_*[Q_i]$ is a moving target
- **Stochastic Approximation Theory**: a partial update to ensure contraction + appropriate step size α implies convergence to the contraction fixed point: $\lim_{i\to\infty} Q_i = Q^*$
- T. Jaakkola, M. Jordan, S. Singh, "On the convergence of stochastic iterative dynamic programming algorithms," Neural computation, 1994.

Batch Gradient Least-Squares Q-Value Iteration

Algorithm Batch Gradient Least-Squares Q-Value Iteration

1: **Init**: $Q_0(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}$ and all $\mathbf{u} \in \mathcal{U}$
2: **loop**
3: $\quad\pi \leftarrow \epsilon$-greedy policy derived from Q_i $\triangleright$ π can be arbitrary!
4: $\quad$Generate episodes $\{\rho^{(k)}\}_{k=1}^{K}$ from π
5: $\quad Q_{i+1} \leftarrow Q_i - \frac{\alpha}{2} \nabla_Q \left[\sum_{k=1}^{K} \sum_{t=0}^{T^{(k)}} \| \mathcal{B}_*[Q_i](\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}, \mathbf{x}_{t+1}^{(k)}) - Q(\mathbf{x}_t^{(k)}, \mathbf{u}_t^{(k)}) \|^2 \right] \Bigg|_{Q=Q_i}$

- Q-learning is a special case with $K = 1$