On-Policy vs Off-Policy Algorithms

CMPT 729 G100

Jason Peng

Overview

- On-Policy vs Off-Policy
- On-Policy Algorithms
- Off-Policy Algorithms
- Trade-Offs

On-Policy vs Off-Policy

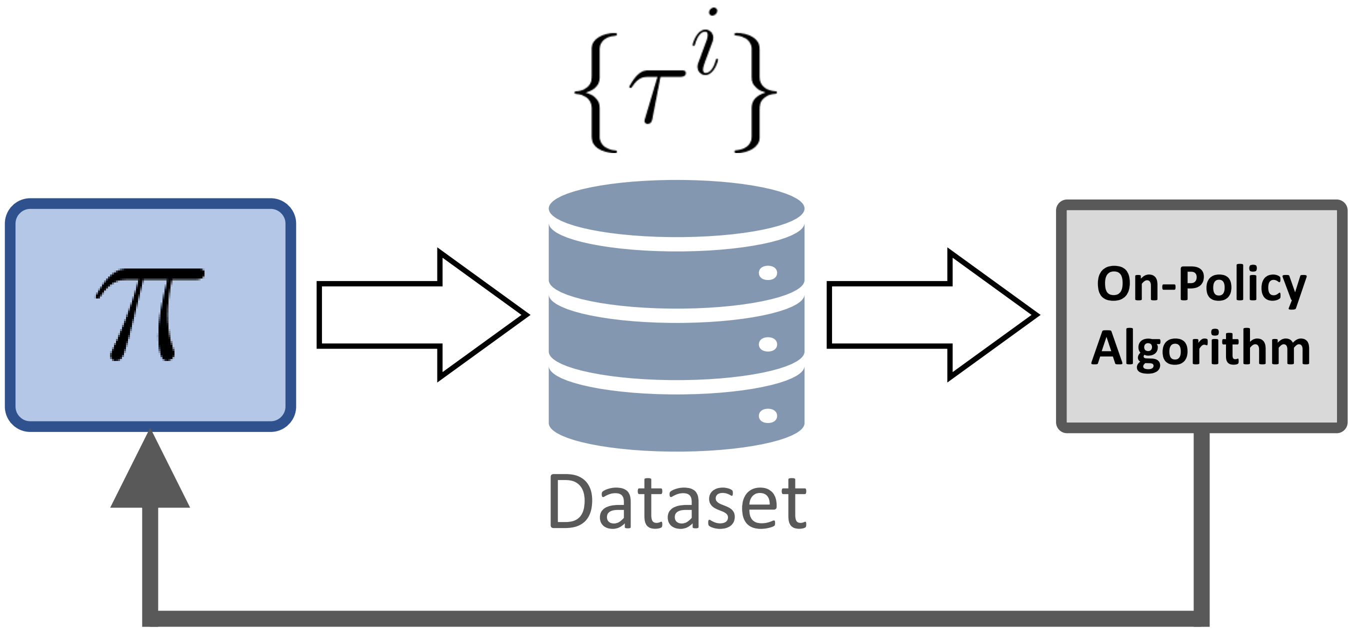
On-policy:

- Model can be update using *only* data collected with the model

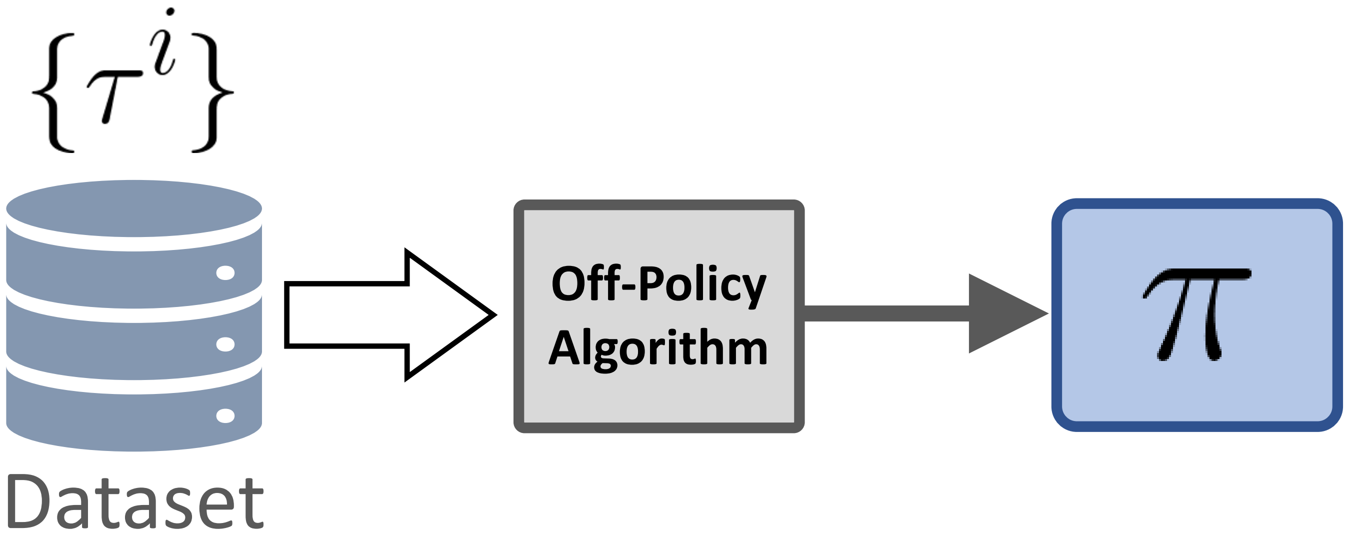
Off-policy:

- Model can be updated using data collected from *other* sources

On-Policy

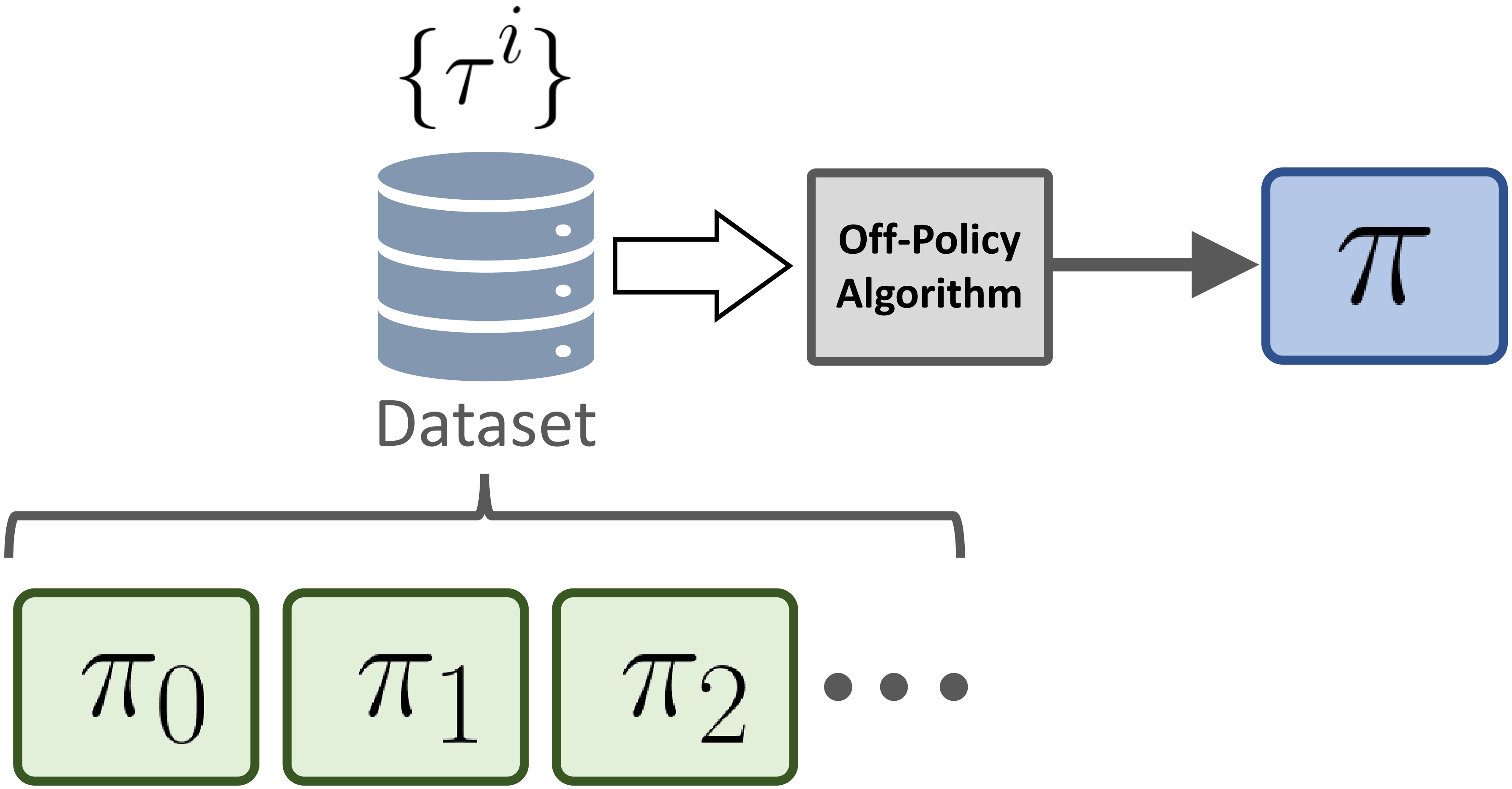


Off-Policy



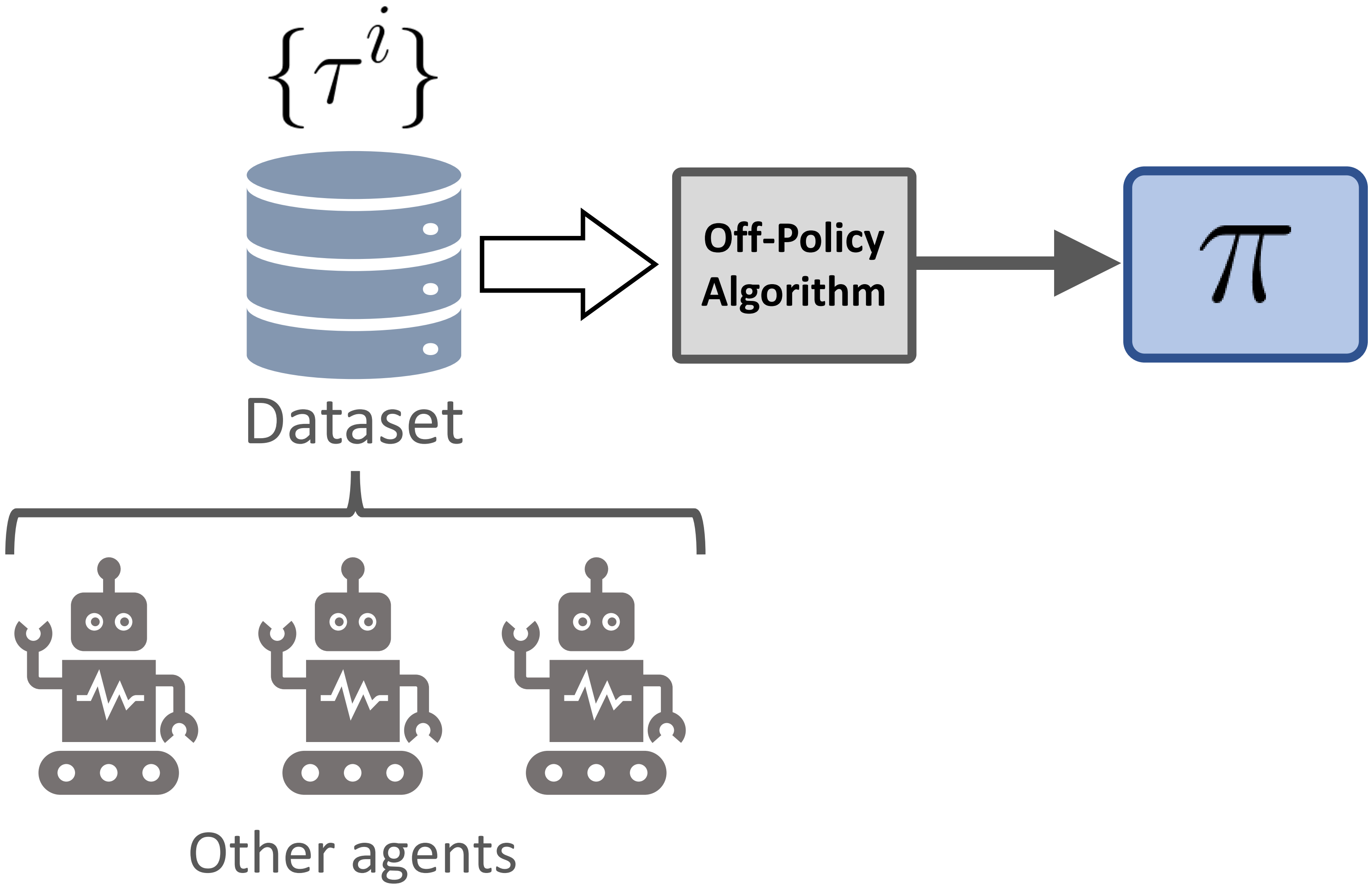
can come from anywhere

Off-Policy

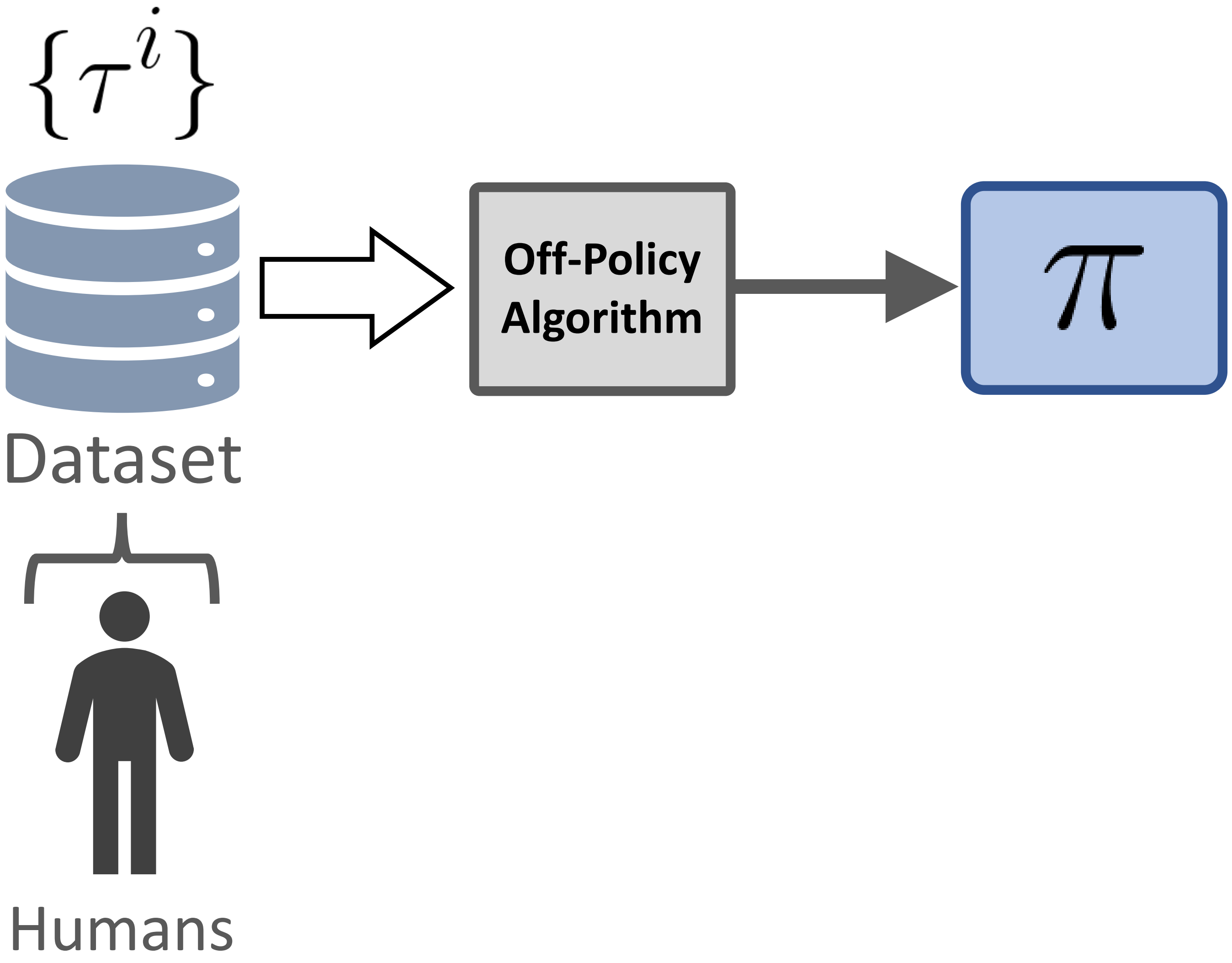


Policies from previous training iterations

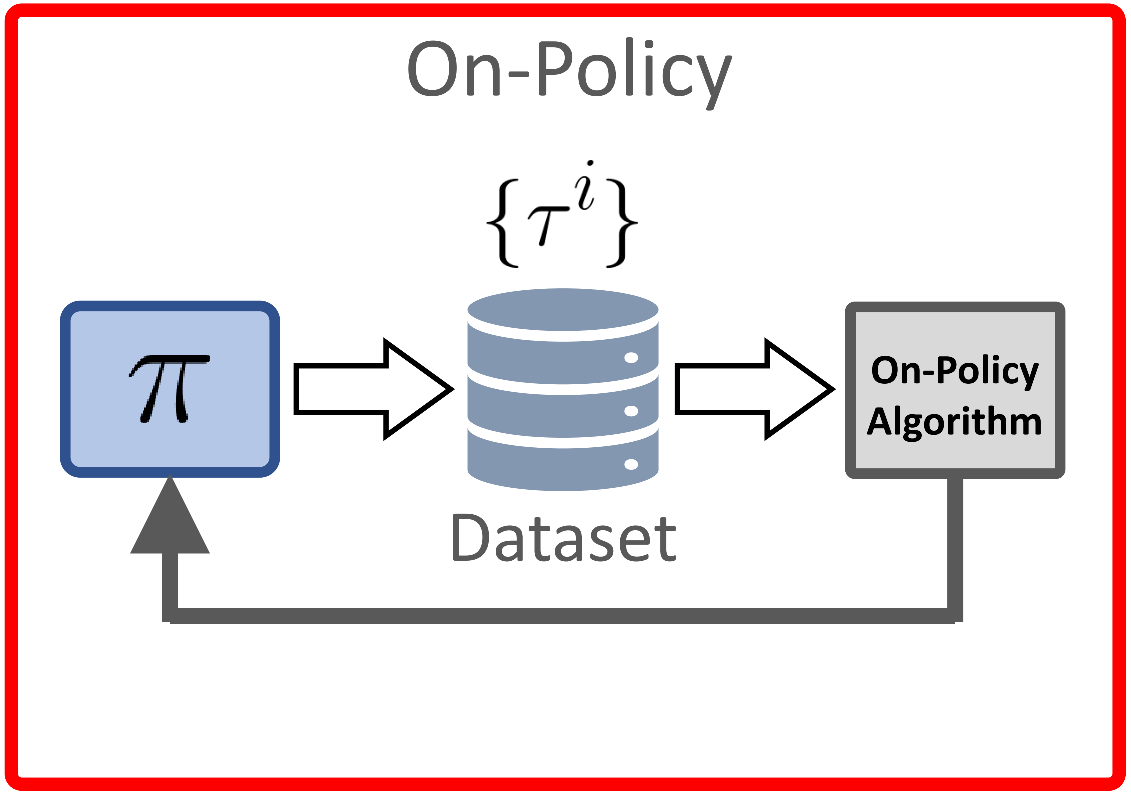
Off-Policy

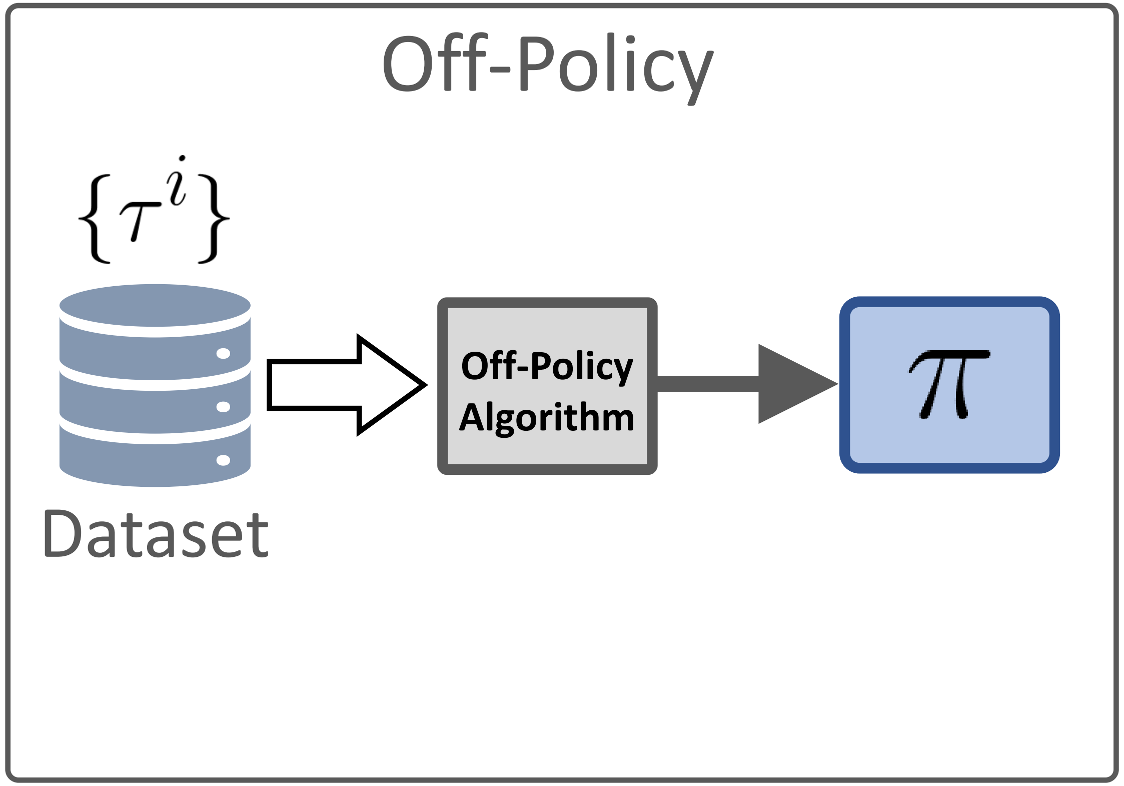


Off-Policy



RL Algorithms





On-Policy (REINFORCE)

$$\nabla_\pi J(\pi) = \underline{\mathbb{E}_{\tau \sim p(\tau|\pi)}} \left[R(\tau) \sum_{t=0}^{T-1} \nabla_\pi \log \pi(\mathbf{a}_t|\mathbf{s}_t) \right]$$

Must be from
current policy

Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning
[Williams 1992]

On-Policy (REINFORCE)

ALGORITHM: REINFORCE

1: $\theta \leftarrow$ initialize policy parameters

2: **while** not done **do**

3: Sample trajectories $\{\tau^i\}$ from policy $\pi_\theta(\mathbf{a}|\mathbf{s})$

4: Estimate policy gradient

$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_i R(\tau^i)\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)$

5: Update policy $\theta \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$

6: **end while**

7: return policy π_θ

Collect data with current policy

Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning [Williams 1992]

On-Policy (REINFORCE)

ALGORITHM: REINFORCE

1: $\theta \leftarrow$ initialize policy parameters

2: **while** not done **do**

3: Sample trajectories $\{\tau^i\}$ from policy $\pi_\theta(\mathbf{a}|\mathbf{s})$

4: Estimate policy gradient
$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N} \sum_i R(\tau^i) \sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)$

5: Update policy $\theta \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$

6: **end while**

7: return policy π_θ

Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning [Williams 1992]

On-Policy (REINFORCE)

ALGORITHM: REINFORCE

1: $\theta \leftarrow$ initialize policy parameters

2: **while** not done **do**

3: Sample trajectories $\{\tau^i\}$ from policy $\pi_\theta(\mathbf{a}|\mathbf{s})$

4: Estimate policy gradient

$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_i R(\tau^i)\sum_t \nabla_\theta \log \pi_\theta(\mathbf{a}_t^i|\mathbf{s}_t^i)$

5: Update policy $\theta \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$

6: **end while**

7: return policy π_θ

Perform just one grad update,
then throw out data

Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning
[Williams 1992]

On-Policy (Policy Gradient)

$$\nabla_\pi J(\pi) = \mathbb{E}_{\mathbf{s}\sim d_\pi(\mathbf{s})}\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\left[\nabla_\pi \log\pi(\mathbf{a}|\mathbf{s})\left(Q^\pi(\mathbf{s},\mathbf{a}) - V^\pi(\mathbf{s})\right)\right]$$

From current policy

Policy Gradient Methods for Reinforcement Learning with Function Approximation
[Sutton et al. 1999]

On-Policy (Policy Gradient)

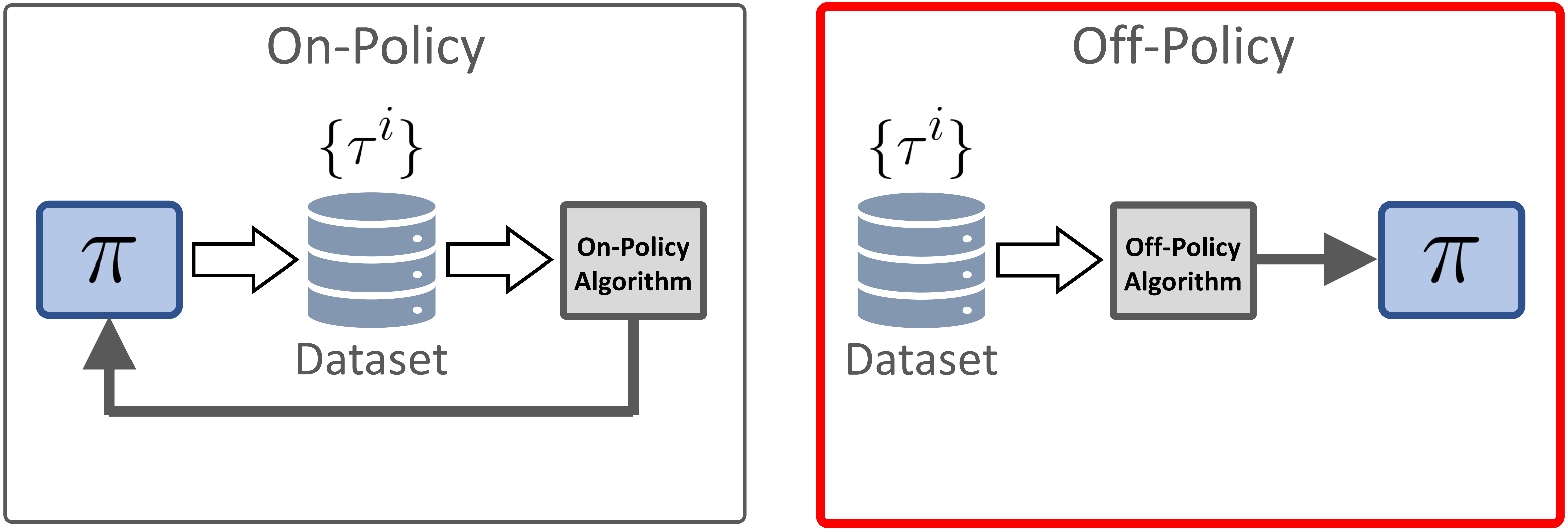
$$\nabla_\pi J(\pi) = \underline{\mathbb{E}_{\mathbf{s}\sim d_\pi(\mathbf{s})}}\,\underline{\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}}\left[\nabla_\pi \log\pi(\mathbf{a}|\mathbf{s})\left(Q^\pi(\mathbf{s},\mathbf{a}) - V^\pi(\mathbf{s})\right)\right]$$

From current policy

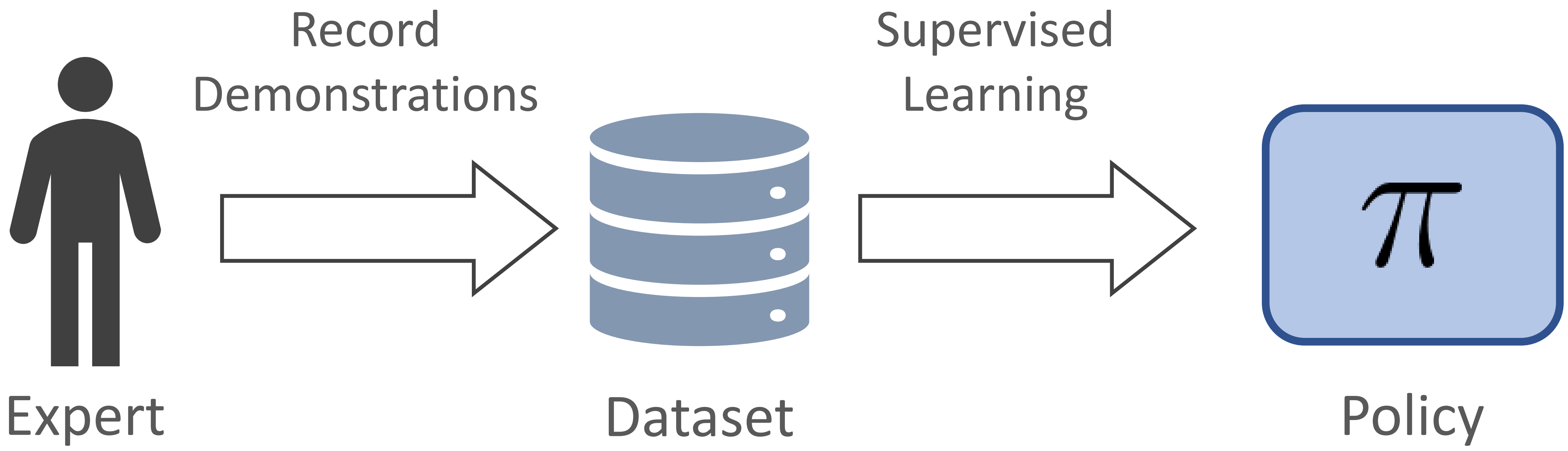
- If data is not from π, PG methods can completely fail to learn anything

Policy Gradient Methods for Reinforcement Learning with Function Approximation [Sutton et al. 1999]

RL Algorithms



Behavioral Cloning

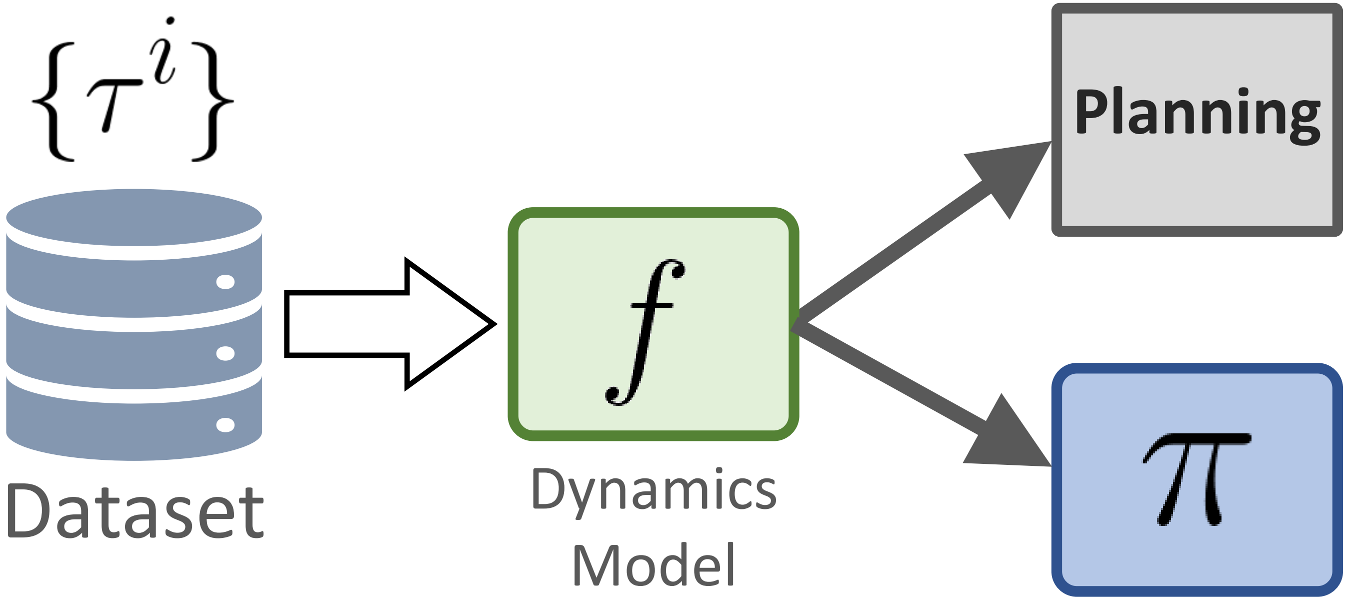


$$\min_{\pi} \; \underline{\mathbb{E}_{(\mathbf{o},\mathbf{a})\sim\mathcal{D}}} \left[-\log \pi(\mathbf{a}|\mathbf{o}) \right]$$

Off-Policy (Model-Based RL)

$$\arg\max_{f} \; \underline{\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}} \left[\log f(\mathbf{s}'|\mathbf{s},\mathbf{a})\right]$$

- Dataset can come from anywhere, as long as it has sufficient coverage of states and actions.



Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\underbrace{\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}', \mathbf{a}')\right)}_{\text{Current Q-function}} - Q(\mathbf{s}, \mathbf{a})\right)^2\right]$$

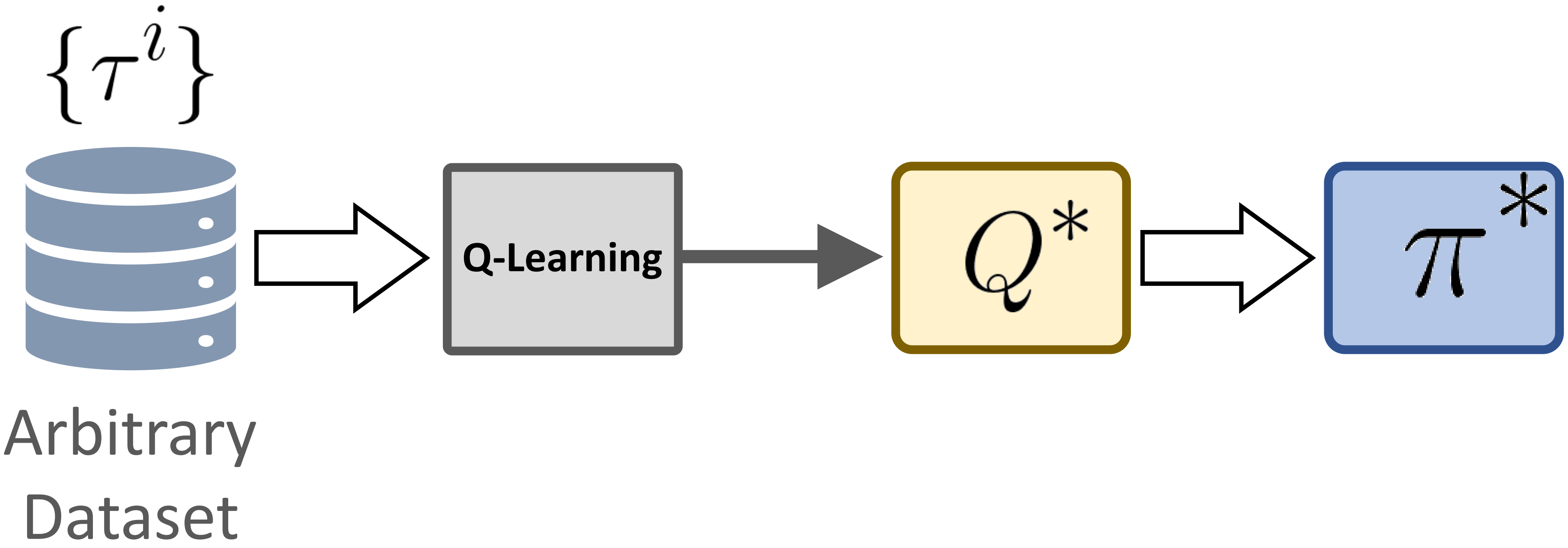
Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \underbrace{\mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}}_{\text{Does not depend on } Q^k\text{!}} \left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}', \mathbf{a}') \right) - Q(\mathbf{s}, \mathbf{a}) \right)^2 \right]$$

- The data distribution can be an arbitrary distribution
- Tabular Q-Learning
  - If dataset covers ***all*** states and actions, then Q-learning will converge to the optimal Q-function
  - Can learn from a completely ***random*** policy, as long as agent observes every state and action at least once

Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}',\mathbf{a}')\right) - Q(\mathbf{s},\mathbf{a})\right)^2\right]$$



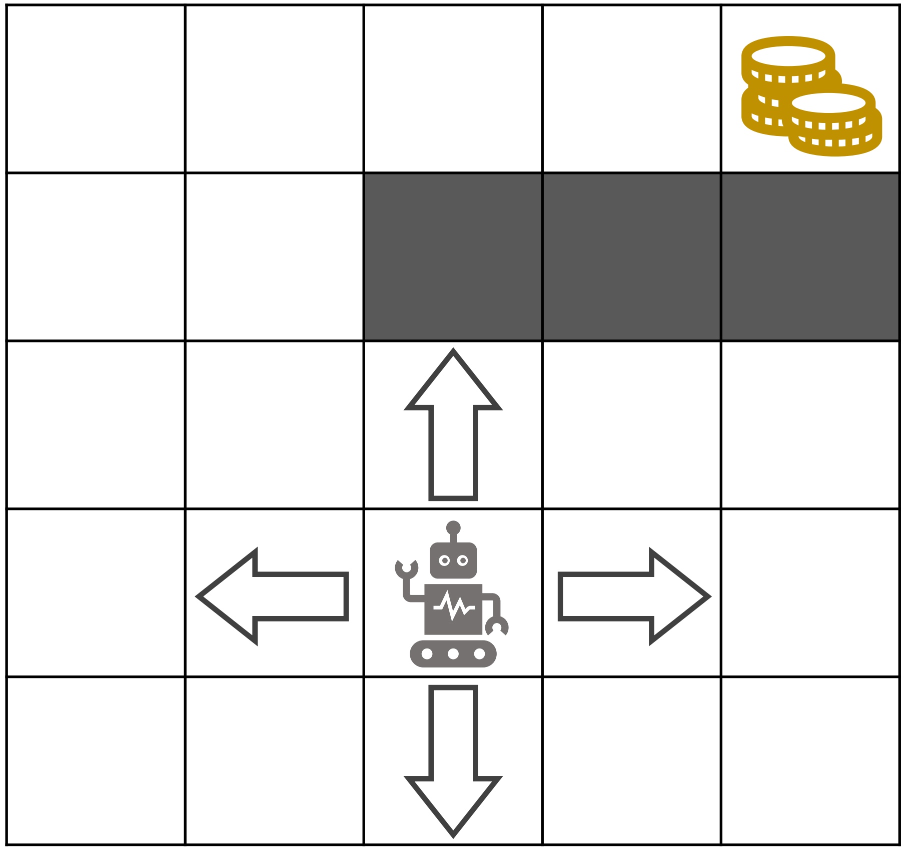
Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}',\mathbf{a}')\right) - Q(\mathbf{s},\mathbf{a})\right)^2\right]$$

- The data distribution can be an arbitrary distribution
- Tabular Q-Learning
  - If dataset covers *all* states and actions, then Q-learning will converge to the optimal Q-function
  - Can learn from a completely *random* policy, as long as agent observes every state and action at least once

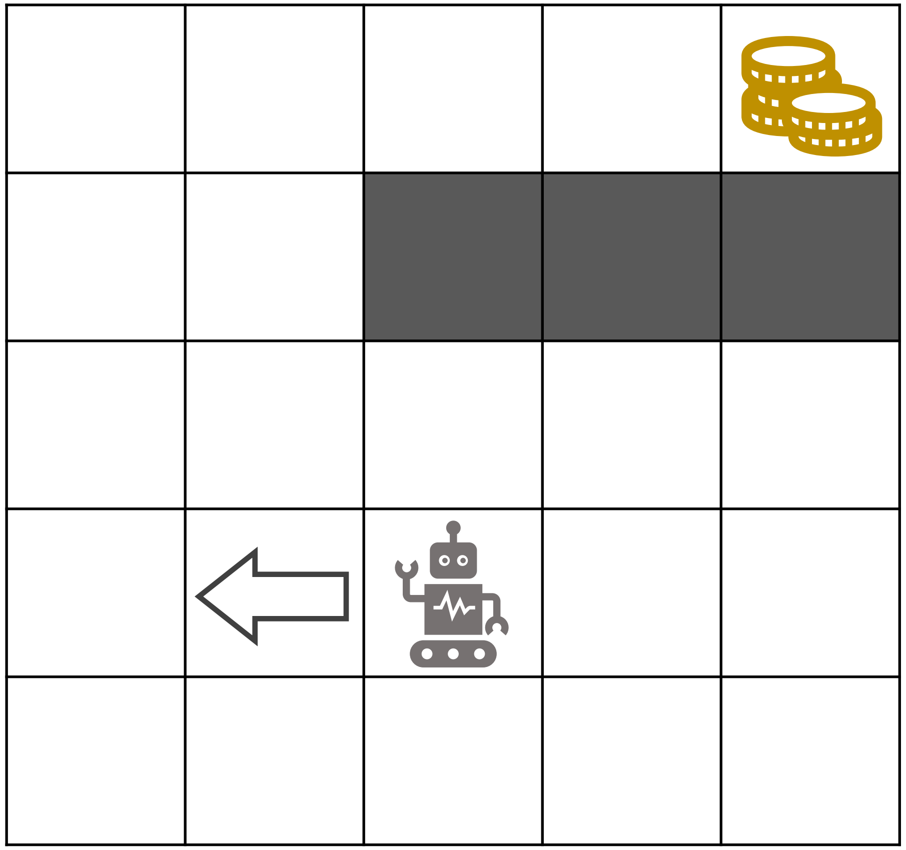
Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}',\mathbf{a}')\right) - Q(\mathbf{s},\mathbf{a})\right)^2\right]$$



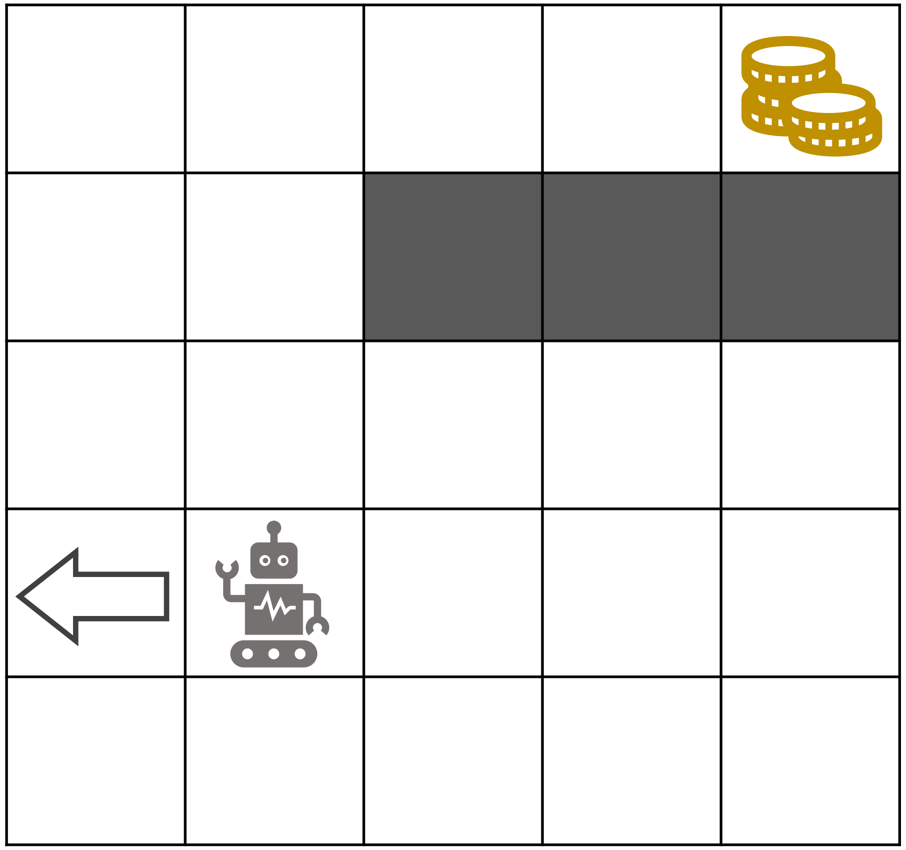
Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}} \left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}',\mathbf{a}') \right) - Q(\mathbf{s},\mathbf{a}) \right)^2 \right]$$



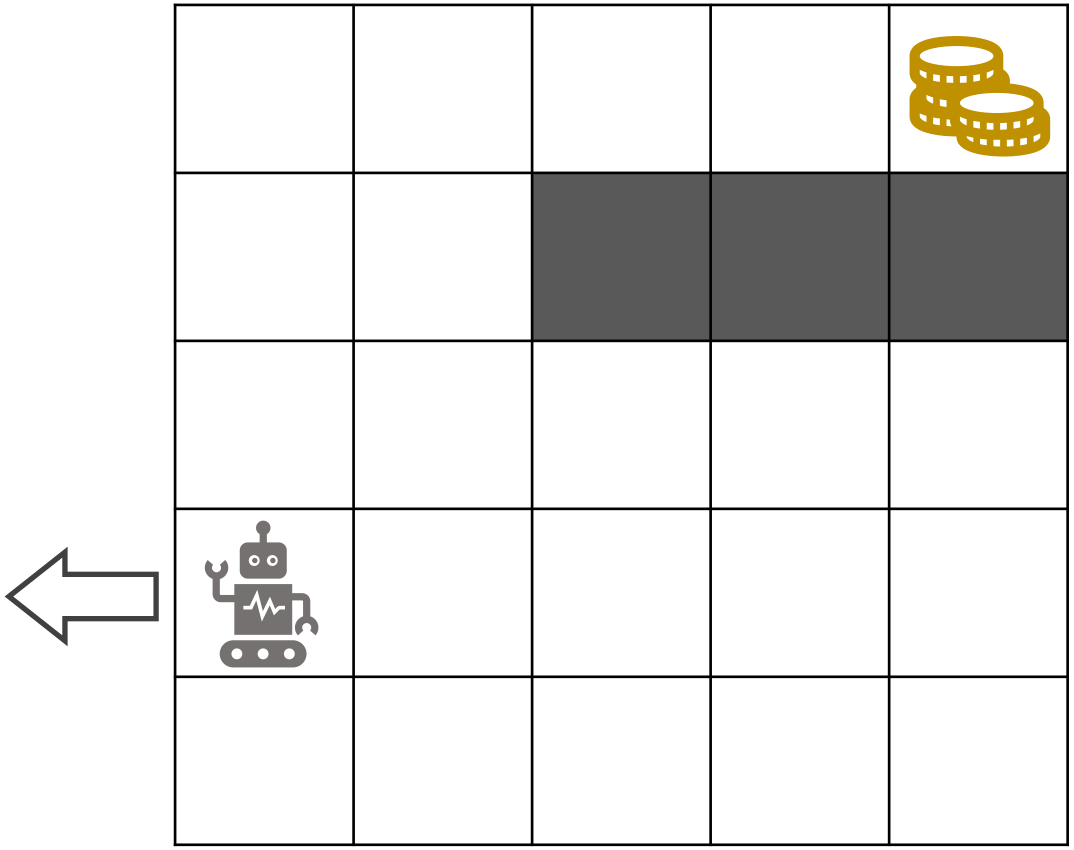
Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}',\mathbf{a}')\right) - Q(\mathbf{s},\mathbf{a})\right)^2\right]$$



Off-Policy (Q-Learning)

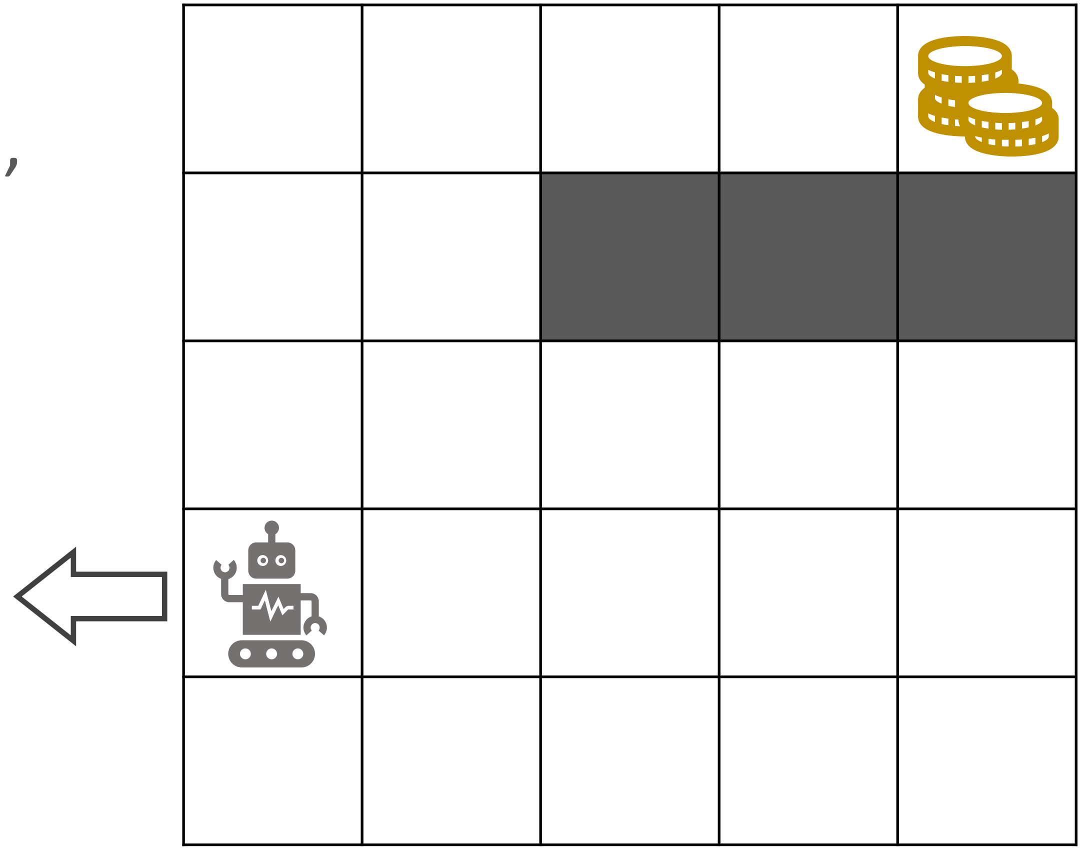
$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}} \left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}',\mathbf{a}') \right) - Q(\mathbf{s},\mathbf{a}) \right)^2 \right]$$



Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}}\left[\left(\left(r+\gamma\max_{\mathbf{a}'}Q^k(\mathbf{s}',\mathbf{a}')\right)-Q(\mathbf{s},\mathbf{a})\right)^2\right]$$

Independent of current model, but the characteristics of the data still matters.



Off-Policy (Q-Learning)

$$Q^{k+1} = \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\sim\mathcal{D}} \left[\left(\left(r + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}', \mathbf{a}') \right) - Q(\mathbf{s}, \mathbf{a}) \right)^2 \right]$$

- The data distribution can be an arbitrary distribution
- Q-Learning + function approximation
  - Not guaranteed to converge to the optimal Q-function
  - Can learn an effective policy from arbitrary dataset with sufficient coverage

Off-Policy (Q-Learning)

ALGORITHM: Q-Learning

$Q^0 \leftarrow$ initialize Q-function
$\mathcal{D} \leftarrow \{\emptyset\}$ initialize dataset

for iteration $k = 0, \ldots, n-1$ **do**
  Sample trajectory τ according to $Q^k(\mathbf{s}, \mathbf{a})$
  Add transitions to dataset $\mathcal{D} = \mathcal{D} \cup \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}'_i)\}$

Keep data from previous iterations for better coverage

  Calculate target values for each sample i:
  $y_i = r_i + \gamma \max_{\mathbf{a}'} Q^k(\mathbf{s}'_i, \mathbf{a}')$

  Update Q-function:
  $Q^{k+1} = \arg\min_Q \; \mathbb{E}_{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{r}_i, \mathbf{s}'_i) \sim \mathcal{D}} \left[(y_i - Q(\mathbf{s}_i, \mathbf{a}_i))^2 \right]$
end for

return Q^n

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s}\sim d_\pi(\mathbf{s})}\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\left[\hat{Q}^\pi(\mathbf{s},\mathbf{a})\right]$$

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s} \sim d_\pi(\mathbf{s})} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \left[\hat{Q}^\pi(\mathbf{s}, \mathbf{a}) \right]$$

$$\nabla_\pi \hat{J}(\pi) \approx \nabla_\pi J(\pi)$$

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s} \sim \boxed{d_\pi(\mathbf{s})}} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \left[\hat{Q}^{\boxed{\pi}}(\mathbf{s}, \mathbf{a}) \right]$$

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s} \sim d_\pi(\mathbf{s})} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \left[\hat{Q}^\pi(\mathbf{s}, \mathbf{a}) \right]$$

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s} \sim \boxed{d_\mu(\mathbf{s})}} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \left[Q^{\boxed{\mu}}(\mathbf{s}, \mathbf{a}) \right]$$

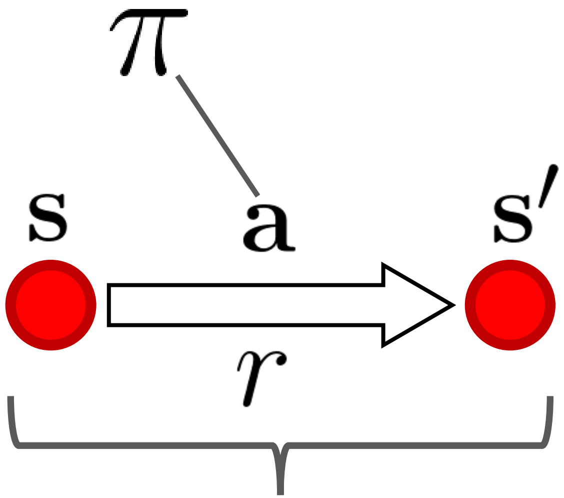
$\mu(\mathbf{a}|\mathbf{s})$: behavior policy

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s}\sim d_\mu(\mathbf{s})}\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\left[Q^\mu(\mathbf{s},\mathbf{a})\right]$$

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s}\sim d_\mu(\mathbf{s})}\mathbb{E}_{\underline{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}}\left[Q^\mu(\mathbf{s},\mathbf{a})\right]$$



π picks action for one step

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s} \sim d_\mu(\mathbf{s})} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \underline{\left[Q^\mu(\mathbf{s}, \mathbf{a}) \right]}$$

π picks action for one step, and then μ takes over

π Is trying to put the agent in a state $\mathbf{s}'$ that μ will be the most effective in.

Off-Policy (SAC)

$$\hat{J}(\pi) = \mathbb{E}_{\mathbf{s} \sim d_\mu(\mathbf{s})} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \left[Q^\mu(\mathbf{s}, \mathbf{a}) \right]$$

- If $\mu(\mathbf{a}|\mathbf{s}) = \pi^*(\mathbf{a}|\mathbf{s})$, then $\pi \to \pi^*$
- If $\mu(\mathbf{a}|\mathbf{s}) \neq \pi^*(\mathbf{a}|\mathbf{s})$, then algorithm will still learn *some* policy, but it might not be optimal
- Will not work for a completely random behavior policy, even if it covers all states and actions

Off-Policy (SAC)

ALGORITHM: SAC

$Q^0 \leftarrow$ initialize Q-function
$\pi^0 \leftarrow$ initialize policy
$\mathcal{D} \leftarrow \{\emptyset\}$ initialize dataset

for iteration $k = 0, ..., n-1$ **do**
  Sample trajectory τ according to $\pi^k(\mathbf{a}|\mathbf{s})$
  Add transitions to dataset $\mathcal{D} = \mathcal{D} \cup \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_i')\}$

  Calculate target values for each sample i:
  $y_i = r_i + \gamma \mathbb{E}_{\mathbf{a}' \sim \pi^k(\mathbf{a}'|\mathbf{s}_i')} \left[Q^k(\mathbf{s}_i', \mathbf{a}') \right]$

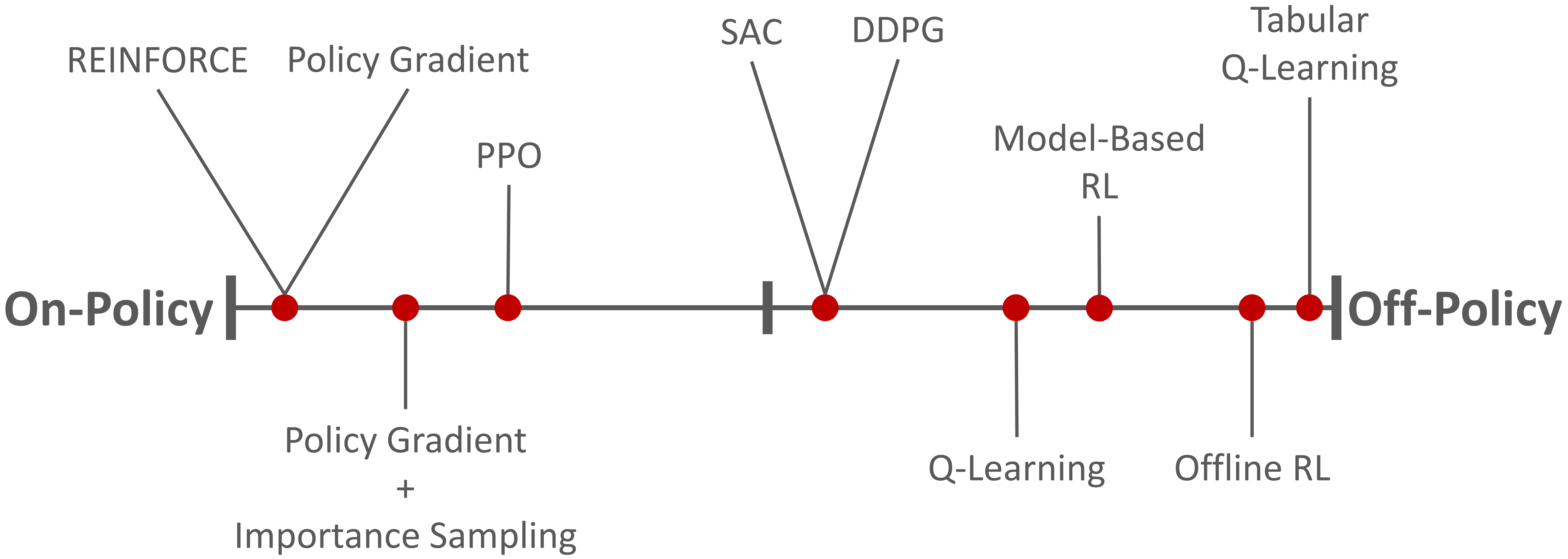
  Update Q-function:
  $Q^{k+1} = \arg\min_Q \; \mathbb{E}_{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{r}_i, \mathbf{s}_i') \sim \mathcal{D}} \left[(y_i - Q(\mathbf{s}_i, \mathbf{a}_i))^2 \right]$

  Update policy:
  $\pi^{k+1} = \arg\max_\pi \; \mathbb{E}_{\mathbf{s}_i \sim \mathcal{D}} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s}_i)} \left[Q^{k+1}(\mathbf{s}_i, \mathbf{a}) \right]$
end for

return π^n

π improves every iteration, and gets closer to π^*

On-Policy vs Off-Policy



Trade-Offs

On-Policy	**Off-Policy**
✗ Sample inefficient	✓ Sample efficient
✓ Fast wall-clock time	✗ Slow wall-lock time
✓ Typically better asymptotic performance	✗ Typically worse asymptotic performance
✓ More stable and easy to tune	✗ More unstable and hard to tune
✗ Exploration limited by action distribution	✓ Flexible exploration

On-Policy Exploration

Exploration: what actions can the policy take?

$$\nabla_\pi J(\pi) = \mathbb{E}_{\mathbf{s} \sim d_\pi(\mathbf{s})} \underline{\mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})}} \left[\nabla_\pi \log \pi(\mathbf{a}|\mathbf{s}) \left(Q^\pi(\mathbf{s}, \mathbf{a}) - V^\pi(\mathbf{s}) \right) \right]$$

On-Policy Exploration

Exploration: what actions can the policy take?

$$\nabla_\pi J(\pi) = \mathbb{E}_{\mathbf{s}\sim d_\pi(\mathbf{s})}\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}\left[\underline{\nabla_\pi \log \pi(\mathbf{a}|\mathbf{s})}\left(Q^\pi(\mathbf{s},\mathbf{a}) - V^\pi(\mathbf{s})\right)\right]$$

Policy Gradient:

- Action distribution must have a differentiable log-likelihood
- Limited to simple action distributions with easy to compute log-likelihoods

Off-Policy Exploration

Exploration: what actions can the policy take?

$$\hat{J}(\pi) = \mathbb{E}_{\underline{\mathbf{s} \sim d_\mu(\mathbf{s})}} \mathbb{E}_{\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})} \left[Q^\mu(\mathbf{s}, \mathbf{a}) \right]$$

SAC:

- Do not need to differentiate behavior policy μ
- Can collect data using any action distribution as long as it has good coverage of actions
- E.g. temporally correlated actions, epsilon-greedy, multi-modal distributions, mixture models, etc.

Summary

- On-Policy vs Off-Policy
- On-Policy Algorithms
- Off-Policy Algorithms
- Trade-Offs