BCQ

Saurabh Gupta

Over estimation of Q-values

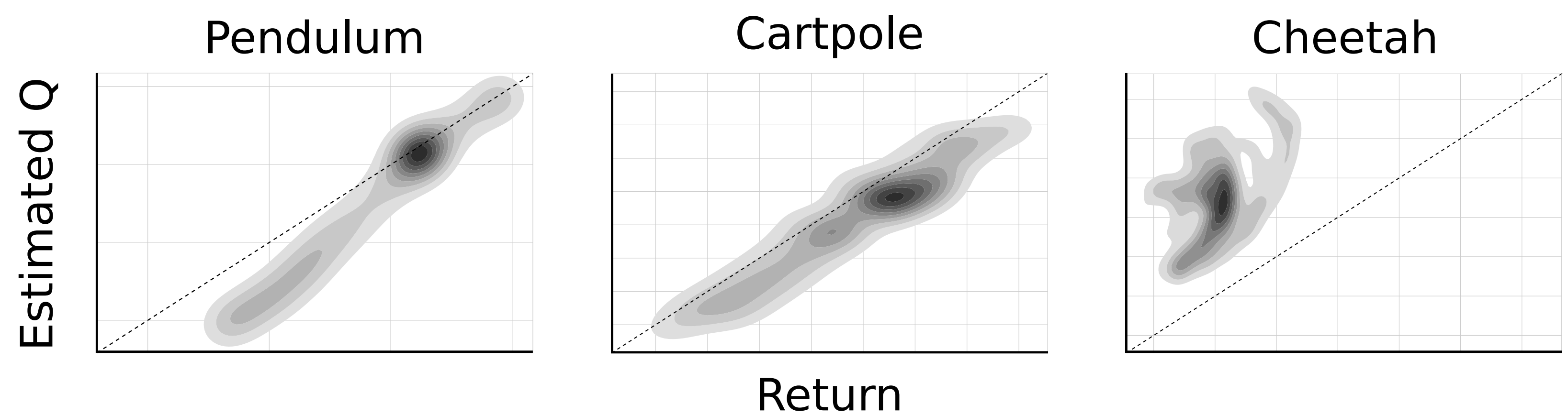
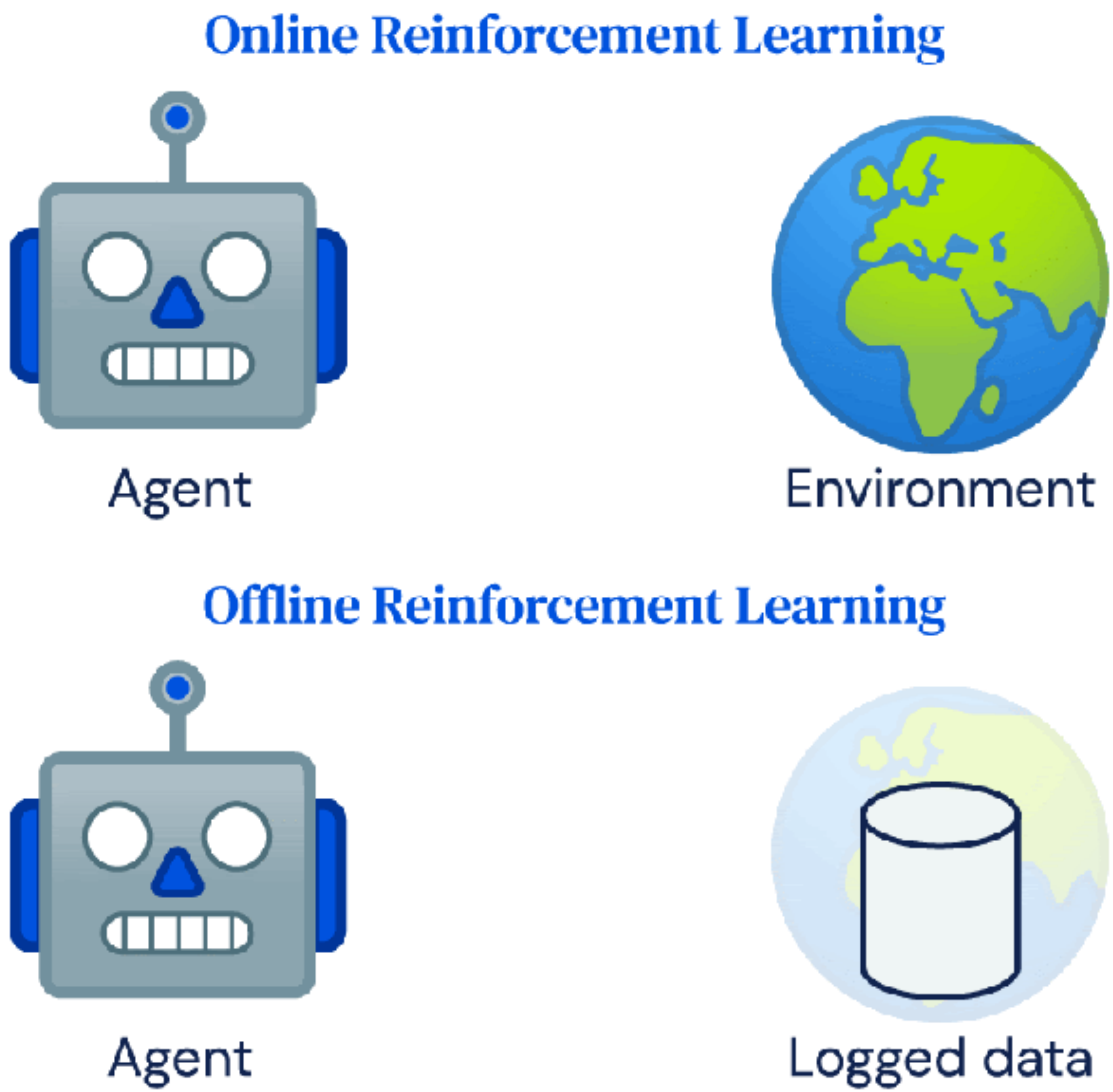


Figure 3: Density plot showing estimated Q values versus observed returns sampled from test episodes on 5 replicas. In simple domains such as pendulum and cartpole the Q values are quite accurate. In more complex tasks, the Q estimates are less accurate, but can still be used to learn competent policies. Dotted line indicates unity, units are arbitrary.

Batch RL / Offline RL

Instead of actively interacting with the environment



Batch RL / Offline RL

Why batch RL?

- Re-use experience: gathering experience is the most expensive part of RL
- Gathering experience may be unsafe
- Learn from other's experience

Problems with Off-line Learning

$$Q(s,a) \leftarrow r + \gamma \max_{a'} Q(s',a') \quad \forall (s,a,s',r) \in \mathcal{D}$$

- *Extrapolation error*
  - we do not know where our estimate of $Q(s',a')$ is good
  - even if we assume $Q(s',a')$ is an unbiased estimate, the $\max$ will cause it to become biased

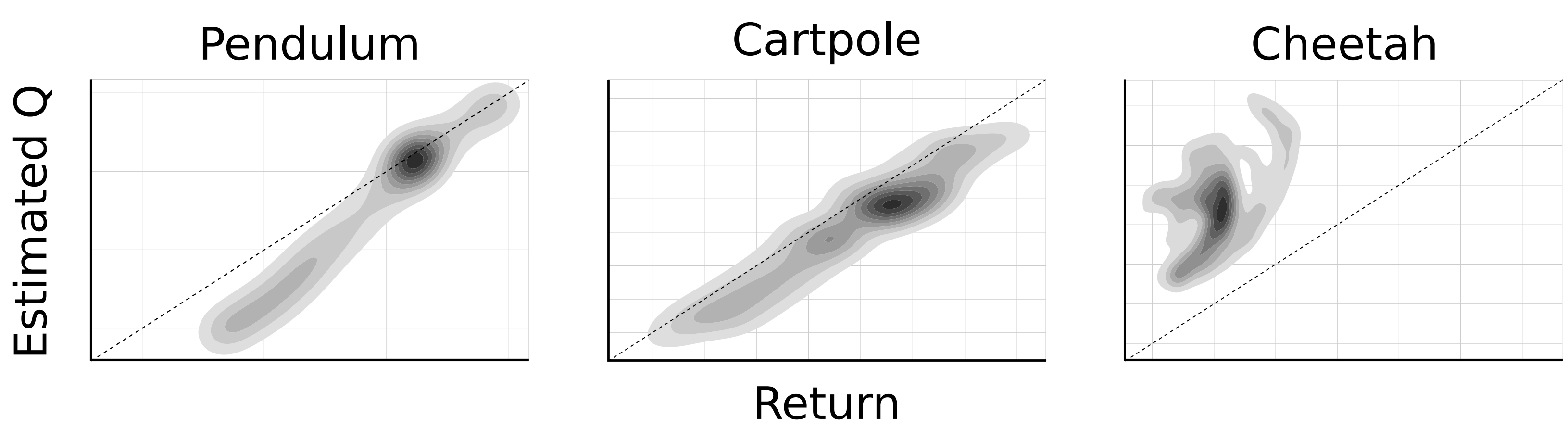
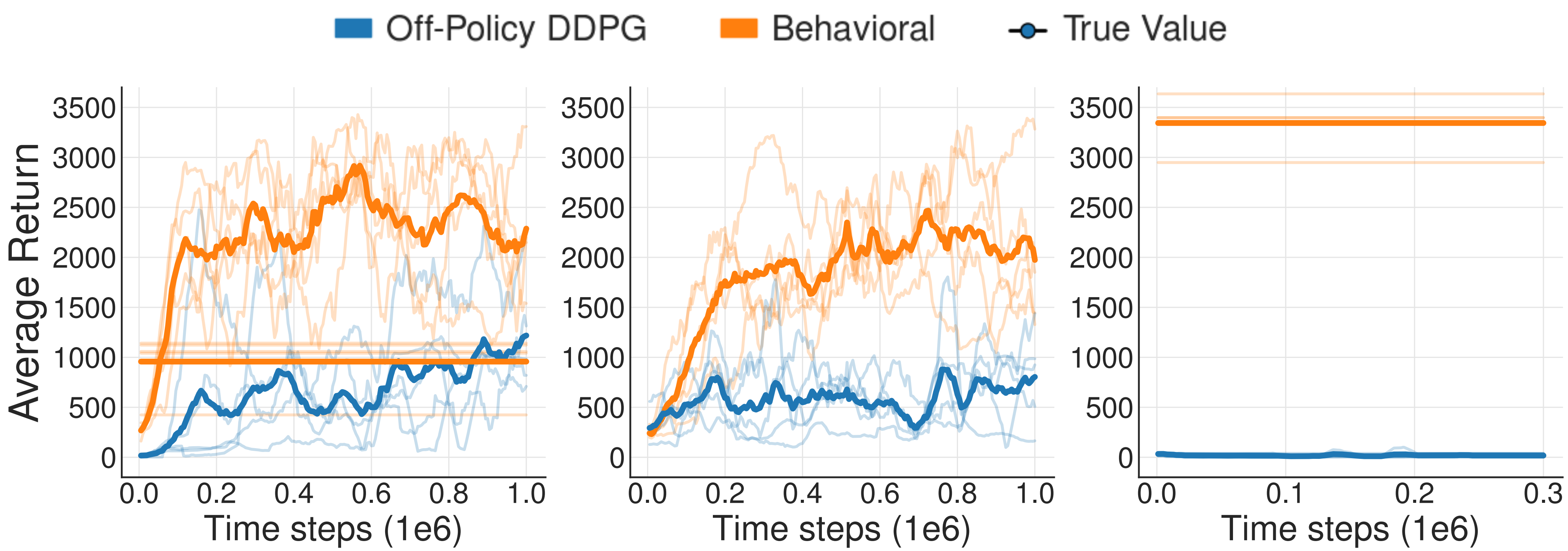


Figure 3: Density plot showing estimated Q values versus observed returns sampled from test episodes on 5 replicas. In simple domains such as pendulum and cartpole the Q values are quite accurate. In more complex tasks, the Q estimates are less accurate, but can still be used to learn competent policies. Dotted line indicates unity, units are arbitrary.

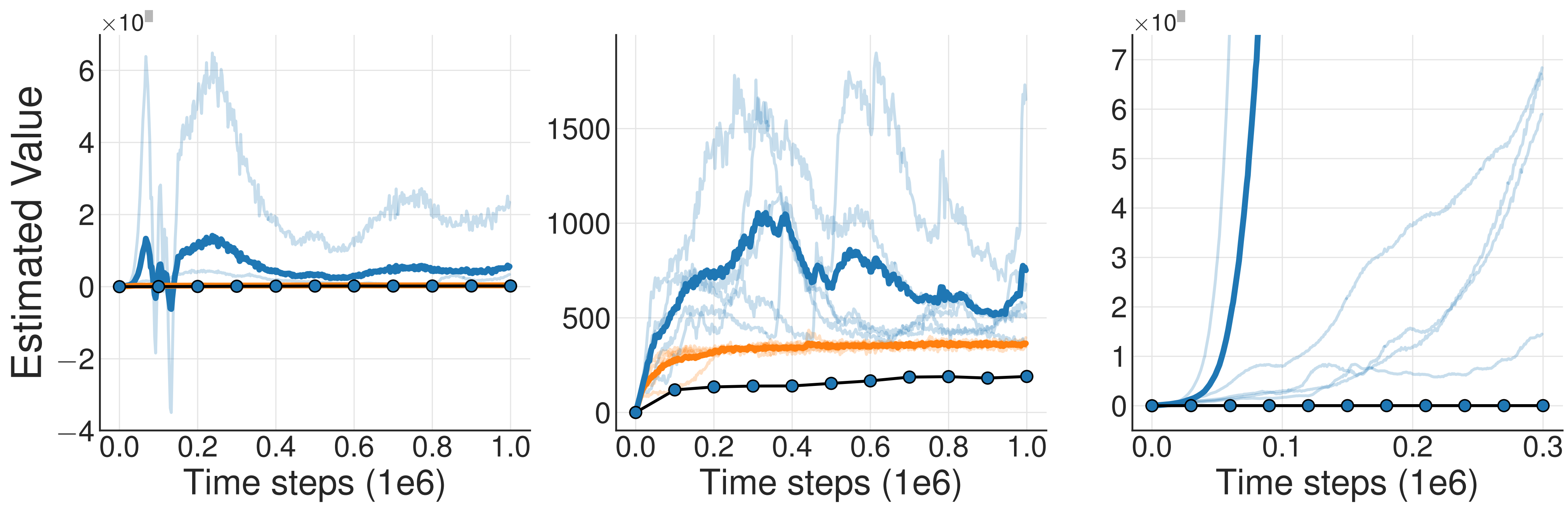
Experiment 1



(a) Final buffer performance

(b) Concurrent performance

(c) Imitation performance



(d) Final buffer value estimate

(e) Concurrent value estimate

(f) Imitation value estimate

But, existing methods work, don't they?

- DQN, DDPG aren't really off-policy, use ϵ-greedy policies
- $\max$ introduces a bias, but unsubstantiated optimism can be tested in subsequent iterations.

Batch-constrained Q-learning

- policy should induce a similar state-action distribution as dataset
    - minimize distance of selected action to data in batch
    - lead to states where familiar data is observed
    - maximize the value function
- train a pair of networks (use minimum of Q-value)

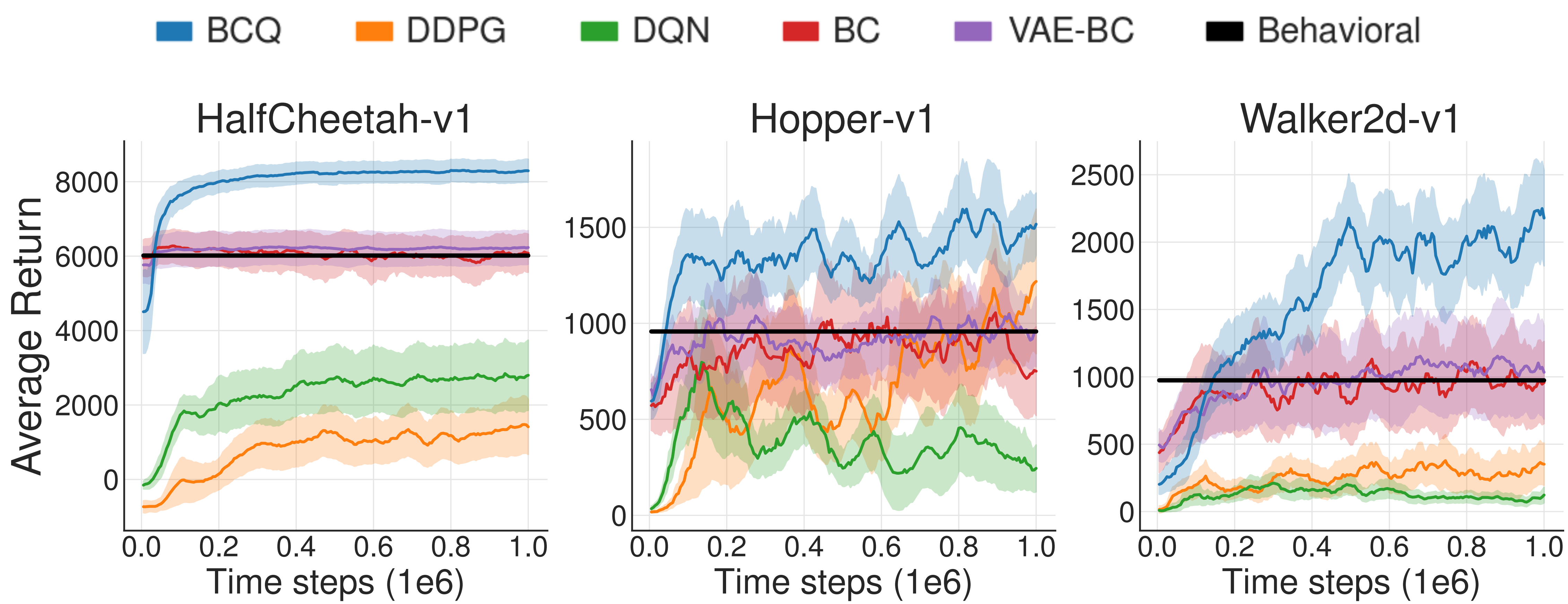
Batch-constrained Q-learning

Algorithm 1 BCQ

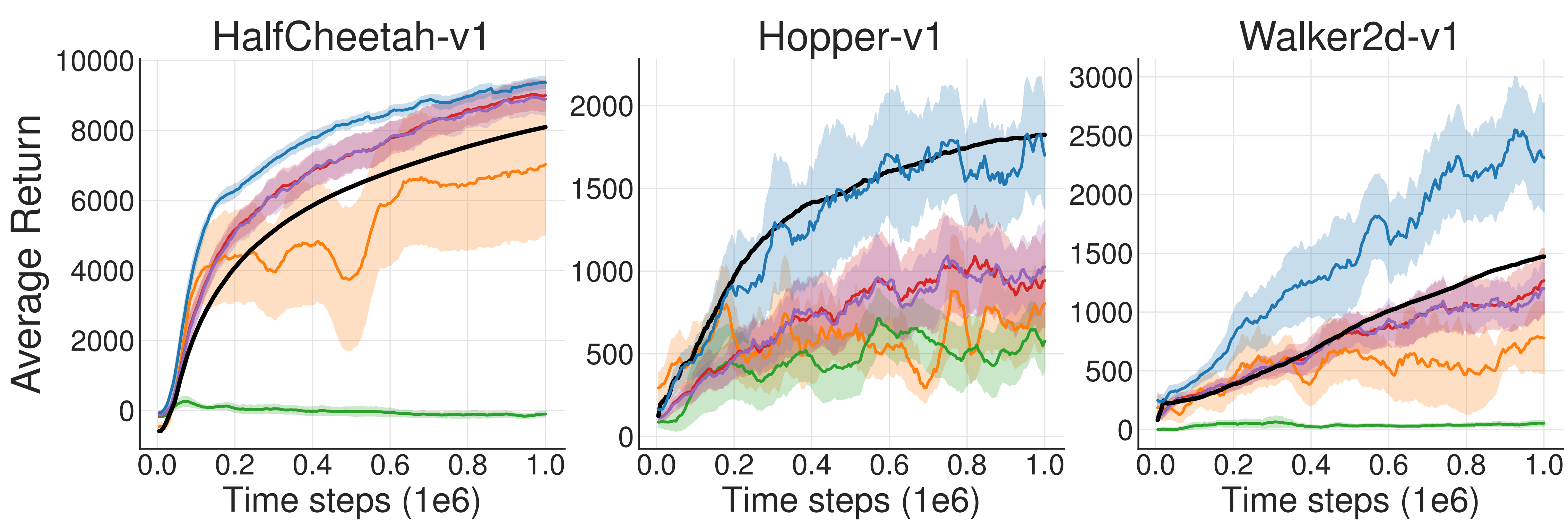
Input: Batch $\mathcal{B}$, horizon T, target network update rate τ, mini-batch size N, max perturbation Φ, number of sampled actions n, minimum weighting λ.
Initialize Q-networks $Q_{\theta_1}, Q_{\theta_2}$, perturbation network ξ_ϕ, and VAE $G_\omega = \{E_{\omega_1}, D_{\omega_2}\}$, with random parameters θ_1, θ_2, ϕ, ω, and target networks $Q_{\theta'_1}, Q_{\theta'_2}$, $\xi_{\phi'}$ with $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2$, $\phi' \leftarrow \phi$.
for $t = 1$ **to** T **do**
  Sample mini-batch of N transitions (s, a, r, s') from $\mathcal{B}$
  $\mu, \sigma = E_{\omega_1}(s, a), \quad \tilde{a} = D_{\omega_2}(s, z), \quad z \sim \mathcal{N}(\mu, \sigma)$
  $\omega \leftarrow \mathrm{argmin}_\omega \sum (a - \tilde{a})^2 + D_{\mathrm{KL}}(\mathcal{N}(\mu, \sigma) || \mathcal{N}(0, 1))$
  Sample n actions: $\{a_i \sim G_\omega(s')\}_{i=1}^n$
  Perturb each action: $\{a_i = a_i + \xi_\phi(s', a_i, \Phi)\}_{i=1}^n$
  Set value target y (Eqn. 13)
  $\theta \leftarrow \mathrm{argmin}_\theta \sum (y - Q_\theta(s, a))^2$
  $\phi \leftarrow \mathrm{argmax}_\phi \sum Q_{\theta_1}(s, a + \xi_\phi(s, a, \Phi)), a \sim G_\omega(s)$
  Update target networks: $\theta'_i \leftarrow \tau\theta + (1 - \tau)\theta'_i$
  $\phi' \leftarrow \tau\phi + (1 - \tau)\phi'$
end for

$$r + \gamma \max_{a_i} \left[\lambda \min_{j=1,2} Q_{\theta'_j}(s', a_i) + (1 - \lambda) \max_{j=1,2} Q_{\theta'_j}(s', a_i) \right] \tag{13}$$

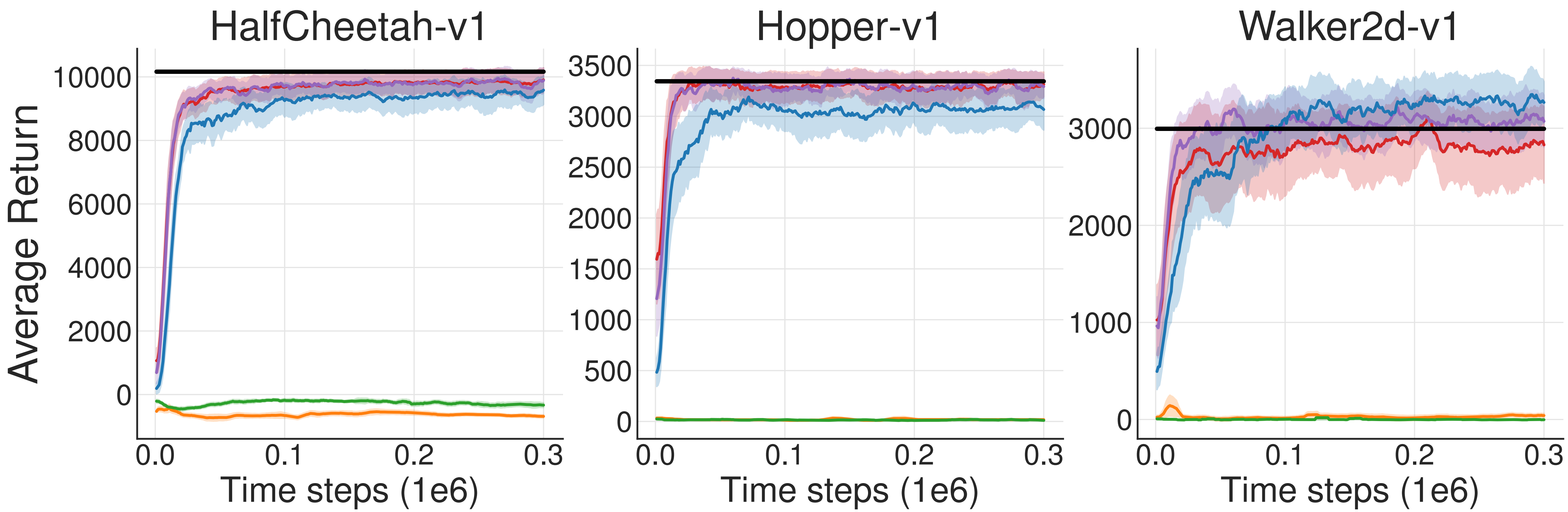
Variational Auto Encoders



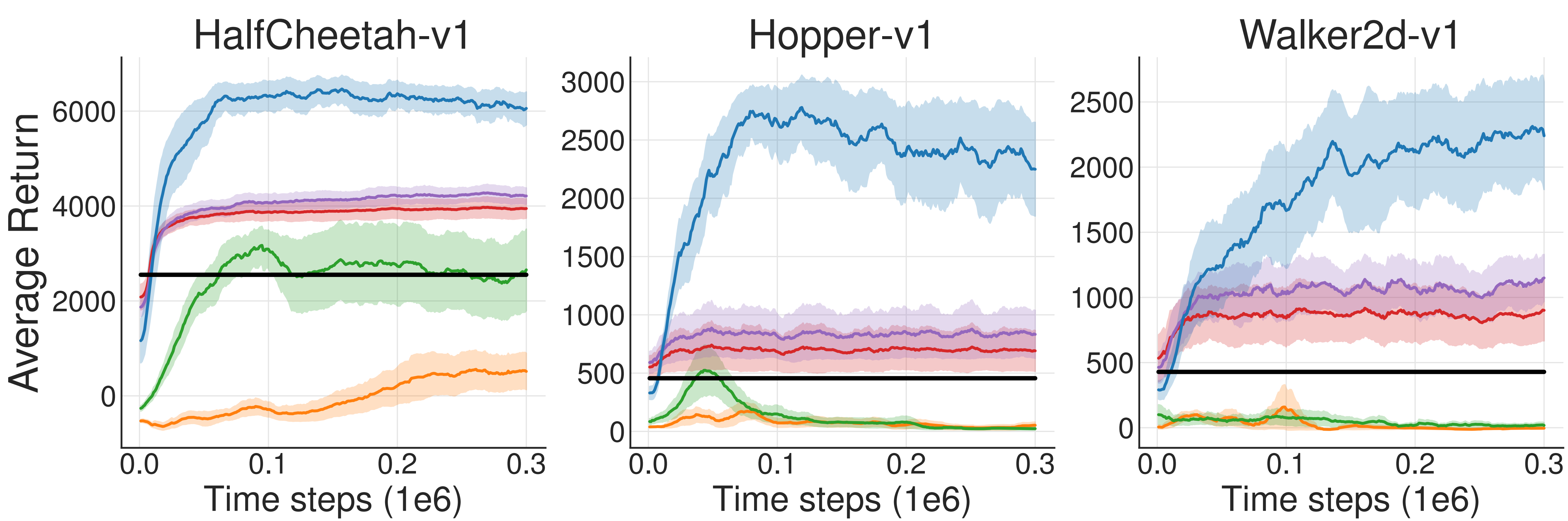
(a) Final buffer performance



(b) Concurrent performance

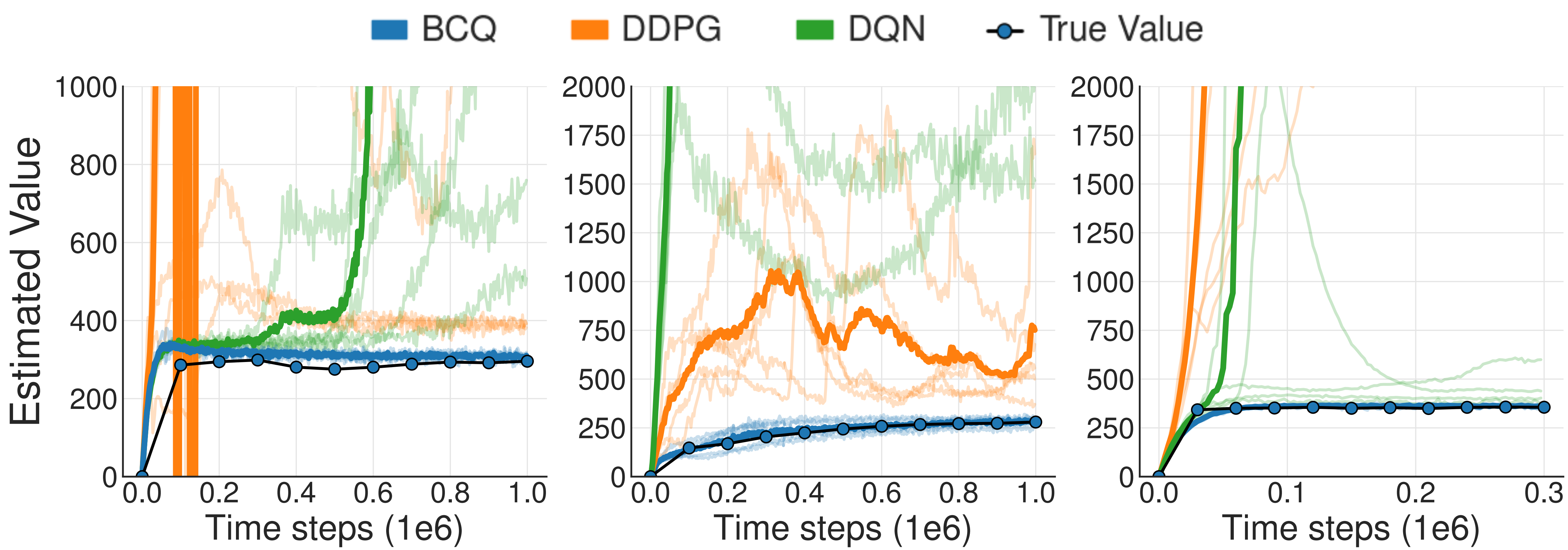


(c) Imitation performance



(d) Imperfect demonstrations performance

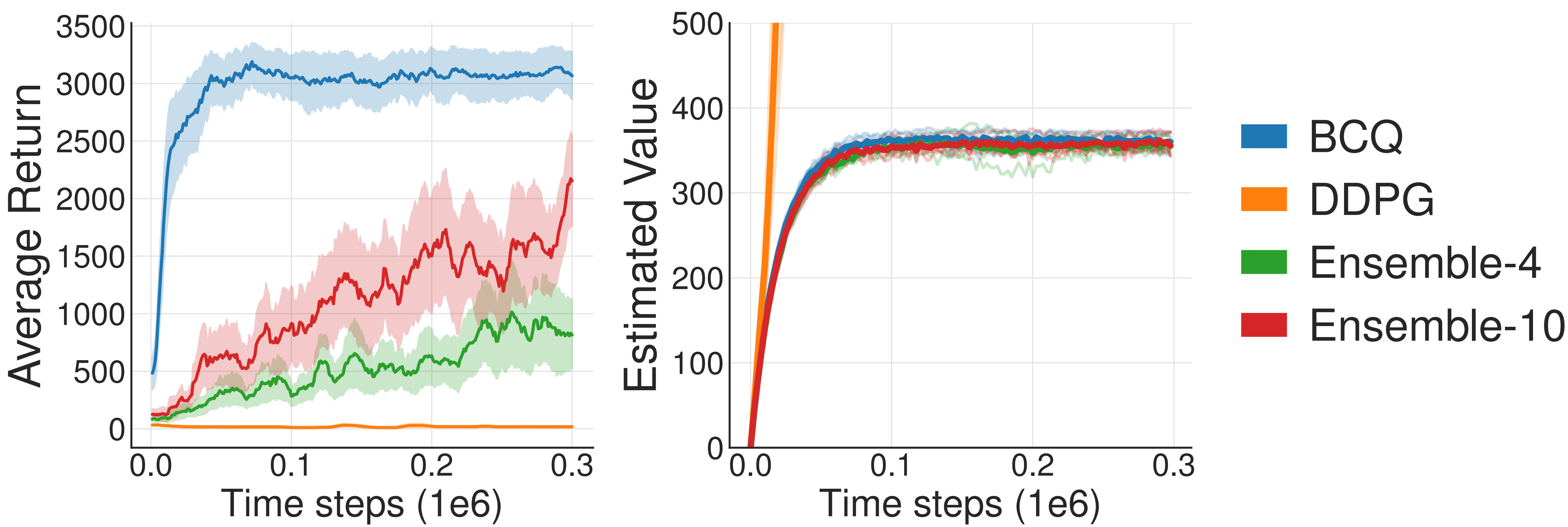
Q-value Estimates



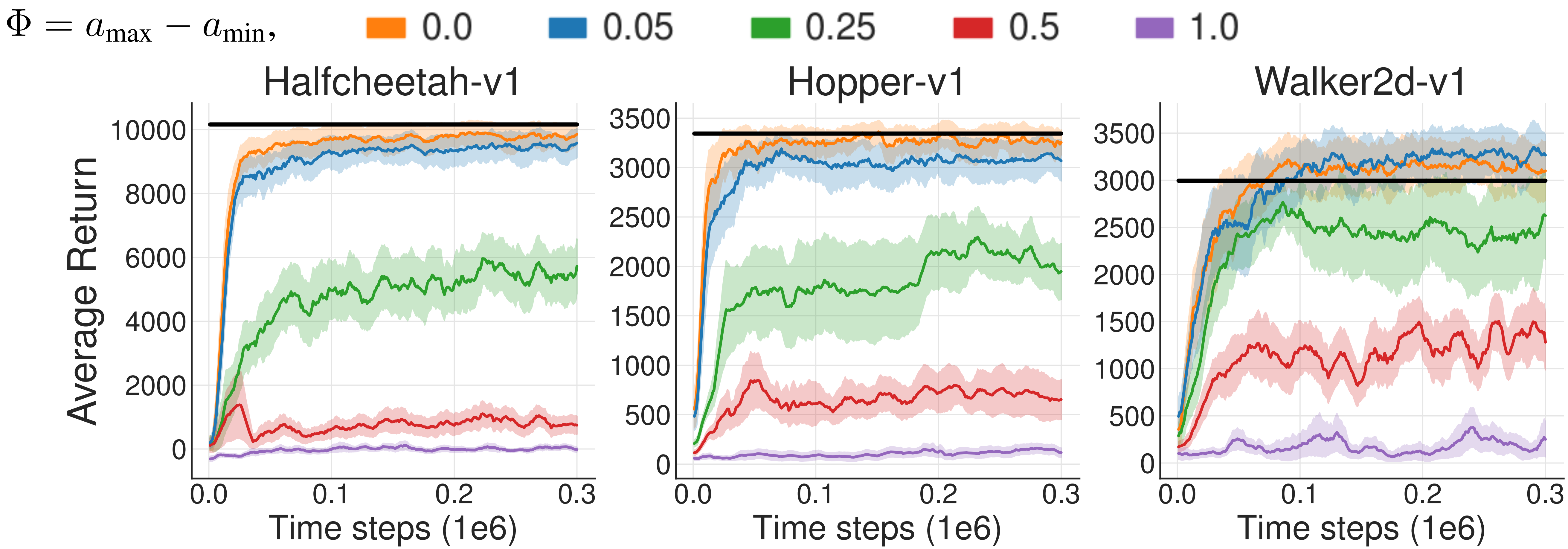
(a) Final Buffer (b) Concurrent (c) Imitation

Related Work

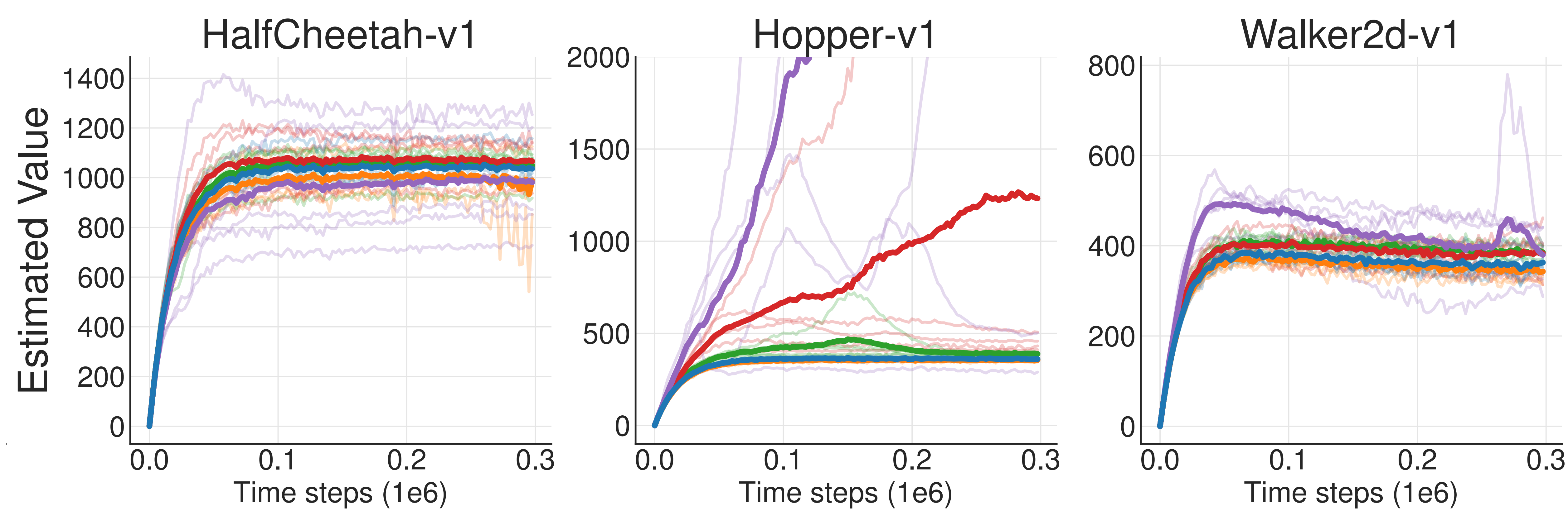
- Modeling uncertainty in neural networks



(a) Imitation performance (b) Imitation value estimates



(a) Imitation performance



(b) Imitation value estimates