468
A MDP Examples

469
A.1 LQR max-following parametric class vs. constituent policies

$$\min_{\{u_t\}_{t=0}^{\infty}} \quad \sum_{t=0}^{\infty} \gamma^t (x_t^T Q x_t + u_t^T R u_t)$$
$$\text{subject to} \quad x_{t+1} = A x_t + B u_t + w_t,$$

To motivate the use of max-following policies in a richer class of MDPs, we consider a traditional control problem with continuous state and action spaces: the discrete linear quadratic regulator. Note that here we analyze the infinite horizon discounted case so that we can analyze the time-invariant value function, but episodic analogues exist. Consider the following setting where $\gamma \in [0,1]$ is a discount factor, and $w_t \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$. Here, we consider the simple case where $Q, R, A = I$ and $B = (1+\epsilon)I$. We know that the optimal policy is of the form $u = -K^* x$ [Bertsekas, 2012] and we set two policies that are only stable along one component and unstable along the other of the form $u_1 = -K_1 x$ and $u_2 = -K_2 x$. It is important to note that the value functions of the individual policies and the optimal policies have exact quadratic forms like $V(x) = x^T P x + q$, but the max-following policy is not necessarily within the same parametric class. For example, P_1 is the solution to the Lyapunov equation $P_1 = (I + K_1^T K_1 + \gamma (A - K_1)^T P_1 (A - K_1))$ and $q_1 = \frac{\gamma}{1-\gamma} \sigma^2 \operatorname{tr}(P_1)$. A similar formula exists for policy 2.

In LQR, for the K_1, K_2 controllers described above, a max-following policy is able to attain higher value than the individual expert policies that have an unstable direction in one axis. Moreover, we see that the optimal policy is obviously superior to all the other policies, but that a max-following policy is more competitive with it than the other individual expert policies. A max-following policy is ultimately able to benefit from the stabilizing component of each axis of the individual policies, which ultimately lets it perform better than any given individual one.

B Additional Proofs

Lemma 4.1 (Worst approximate max-following policy competes with best fixed policy). *For any $\varepsilon \in (0,1]$ and any episode length H, let $\beta \in \Theta(\frac{\varepsilon}{H})$. Then for any MDP $\mathcal{M}$ with starting state distribution μ_0, and any K policies Π^k defined on $\mathcal{M}$,*

$$\min_{\pi \in \Pi_\beta^{k^*}} \mathbb{E}_{s_0 \sim \mu_0} \left[V^{\hat{\pi}}(s_0) \right] \geq \max_{k \in [K]} \mathbb{E}_{s_0 \sim \mu_0} \left[V^k(s_0) \right] - O(\varepsilon).$$

Proof. We will prove the claim inductively, showing that for all $C \in [H]$, if we run any approximate max-following policy for C steps, and then continue following the policy π^k chosen at step C for the rest of the episode, then our expected return is not much worse than if we had followed any fixed π^k for the whole episode.

Somewhat more formally, recalling the definition of the set of approximate max-following policies $\Pi_\beta^{k^*}$ (Definition 2.3), at every time $h \in [H]$ and state $s \in \mathcal{S}$, a policy $\pi \in \Pi_\beta^{k^*}$ takes action $\pi_h^t(s)$ for a $\pi^t \in \Pi^k$ such that $V_h^t(s) \geq \max_{k \in [K]} V_h^k(s) - \beta$. Letting $\pi^{t(s,h)}$ denote the $\pi^t \in \Pi^k$ that π follows at state s and time h, we will show that if at some step $C \in [H]$ we have

$$\mathbb{E}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + \sum_{h=C+1}^{H-1} R(s_h, \pi_h^{t(s_C, C)}(s_h)) \right] \geq \max_{k \in [K]} \mathbb{E}_{s_0 \sim \mu_0} \left[V^k(s_0) \right] - O(\tfrac{\varepsilon(C+1)}{H}),$$

for all $\pi \in \Pi_\beta^{k^*}$, then the same holds for $C+1$ for all π.

In the base case, $C = 0$, the claim

$$\mathbb{E}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{H-1} R(s_h, \pi_h^{t(s_0, 0)}(s_h)) \right] \geq \max_{k \in [K]} \mathbb{E}_{s_0 \sim \mu_0} \left[V^k(s_0) \right] - O(\tfrac{\varepsilon}{H})$$

13

for all $\pi \in \Pi_\beta^{k^*}$ and all $\pi^k \in \Pi^k$, follows straightforwardly from the definition of $\Pi_\beta^{k^*}$ and setting of $\beta \in \Theta(\frac{\varepsilon}{H})$, since

$$\mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{H-1} R(s_h, \pi_h^{t(s_0,0)}(s_h)) \right] = \mathop{\mathbb{E}}_{s_0 \sim \mu_0} [V^{\pi^{t(s_0,0)}}(s_0)]$$

$$\geq \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[\max_{k \in [K]} V^k(s_0) - O(\tfrac{\varepsilon}{H}) \right]$$

$$\geq \max_{k \in [K]} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^k(s_0) \right] - O(\tfrac{\varepsilon}{H}).$$

We now prove the inductive step. We wish to show that if at step C, we have for some $\pi \in \Pi_\beta^{k^*}$

$$\mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + \sum_{h=C+1}^{H-1} R(s_h, \pi_h^{t(s_C,C)}(s_h)) \right] \geq \max_{k \in [K]} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^k(s) \right] - O(\tfrac{\varepsilon(C+1)}{H}),$$

then continuing to follow π at step $C+1$ and following $\pi^{t(s_{C+1},C+1)}$ thereafter reduces expected return by $O(\frac{\varepsilon}{H})$. Now if $\pi_{C+1}(s_{C+1}) = \pi_{C+1}^t(s_{C+1})$ for $\pi^t \in \Pi^k$, it must be the case that

$$V_{C+1}^t(s_{C+1}) \geq \max_{k \in [K]} V_{C+1}^k(s_{C+1}) - O(\tfrac{\varepsilon}{H}),$$

otherwise $\pi \notin \Pi_\beta^{k^*}$. It follows that

$$\mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C+1} R(s_h, \pi_h(s_h)) + \sum_{h=C+2}^{H-1} R(s_h, \pi_h^{t(s_{C+1},C+1)}(s_h)) \right]$$

$$= \mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + V_{C+1}^{t(s_{C+1},C+1)}(s_{C+1}) \right] \quad \text{\small (by definition of V and $\pi_{C+1}(s_{C+1})$)}$$

$$\geq \mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + \max_{k \in [K]} V_{C+1}^k(s_{C+1}) - O(\tfrac{\varepsilon}{H}) \right] \quad \text{\small (from $\pi \in \Pi_\beta^{k^*}$)}$$

$$\geq \mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + V_{C+1}^{t(s_C,C)}(s_{C+1}) - O(\tfrac{\varepsilon}{H}) \right]$$

$$= \mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + \sum_{h=C+1}^{H-1} R(s_h, \pi_h^{t(s_C,C)}(s_h)) \right] - O(\tfrac{\varepsilon}{H}) \quad \text{\small (by definition of V)}$$

$$\geq \max_{k \in [K]} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^k(s) \right] - O(\tfrac{\varepsilon(C+2)}{H}) \quad \text{\small (by inductive hypothesis)}$$

and so the claim holds for time $C+1$, for any $\pi \in \Pi_\beta^{k^*}$ for which it holds for time C. We showed the base case $C = 0$ hold for all $\pi \in \Pi_\beta^{k^*}$, and therefore we have

$$\mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) + \sum_{h=C+1}^{H-1} R(s_h, \pi_h^{t(s_C,C)}(s_h)) \right] \geq \max_{k \in [K]} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^k(s) \right] - O(\tfrac{\varepsilon(C+1)}{H})$$

for all $C \in [H]$. In particular, for $C = H - 1$ we conclude that

$$\mathop{\mathbb{E}}_{s_0 \sim \mu_0, P} \left[\sum_{h=0}^{C} R(s_h, \pi_h(s_h)) \right] \geq \max_{k \in [K]} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^k(s) \right] - O(\varepsilon)$$

and it follows that

$$\min_{\pi \in \Pi_\beta^{k^*}} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^{\hat\pi}(s_0) \right] \geq \max_{k \in [K]} \mathop{\mathbb{E}}_{s_0 \sim \mu_0} \left[V^k(s_0) \right] - O(\varepsilon).$$

$\square$

14

C Additional information about experiments

For our experiments, we use a heuristic version of MaxIteration that operates in rounds. First, the algorithm collects a set of trajectories using every policy to initialize the respective value functions. Then, in every round the algorithm for every policy exectues the max-following policy for β steps and the switches to the respective constituent policy. At the end of each round, value functions of constituent policies are updated. β is uniformly spaced along the full horizon and thus, depends on the number of rounds and the horizon. The total number of episodes is an upper bound on the number of samples collected which is what we determine to compare run-times between MaxIteration and IQL. Finally, we use a γ discounting which has been shown to have regularizing effects on the value function updates [Amit et al., 2020].

For IQL, we use the d3rlpy implementations [Seno and Imai, 2022] and code provided by Hussing et al. [2023].

C.1 Hyperparameters

Both algorithms are run for $10'000$ steps initially (to initialize value functions for MaxIteration and to pre-fill the buffer for IQL) before doing updates and then for $50'000$ steps for online training.

All neural networks use ReLU [Glorot et al., 2011] Multi-layer perceptrons with 2 layers and a hidden dimension of 256 per layer.

Table 1: Hyperparameters for MaxIteration

Optimizer	Adam
Adam β_1	0.9
Adam β_2	0.999
Adam ε	$1e-8$
Value Function Learning Rate	$1e-4$
Number of rounds	50
Number of gradient steps per round	40'000
Batch Size	64
γ	0.99

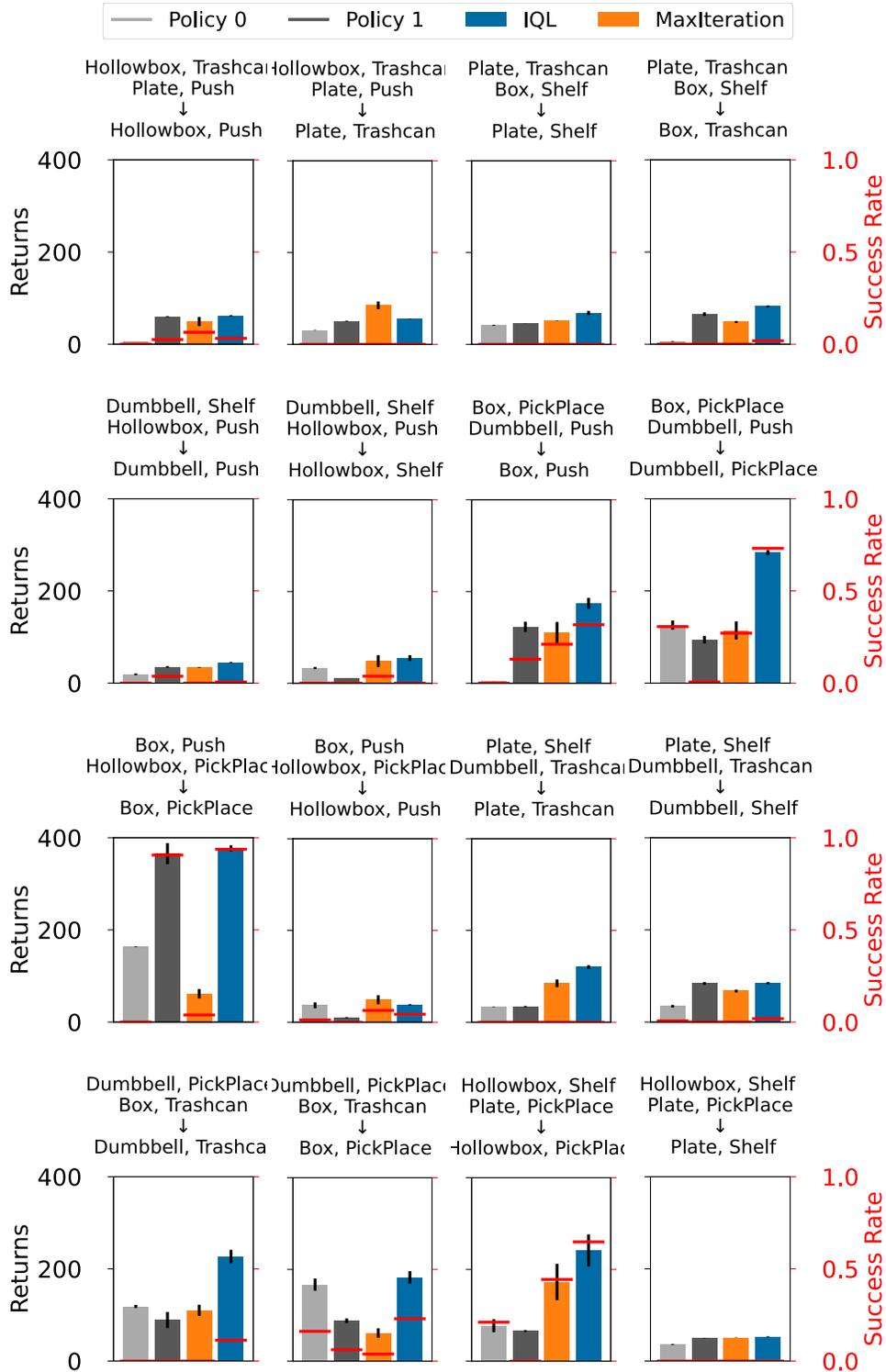
Table 2: Hyperparameters for Implicit Q-Learning

Optimizer	Adam
Adam β_1	0.9
Adam β_2	0.999
Adam ε	$1e-8$
Actor Learning Rate	$4e-3$
Critic Learning Rate	$4e-3$
Batch Size	#Tasks $\times 256$
n_steps	1
γ	0.99
τ	0.005
n_critics	2
expectile	0.7
weight_temp	3.0
max_weight	100

15

530 **C.2 Full results on CompoSuite**

Figure 4

16

C.3 Results on DM Control

We run our MaxIteration algorithm on the DM Control benchmarks [Tunyasuvunakool et al., 2020] similar to the MAPS [Liu et al., 2023] setup. In their setup, the constituent policies correspond to different 3 checkpointed models in one run of the online Soft-Actor critic [Haarnoja et al., 2018] algorithm. As a result, it is generally true that the latest checkpointed model will outperform the previous two checkpoints meaning one constituent policy is strictly better everywhere than the others. We report the final performance over 5 seeds using 16 evaluation trajectories in Figure 5. The results show that our algorithm behaves as expected and always uses the best oracle. Without policy improvement operator, this setup does not allow us to exceed the performance of the constituent policies.

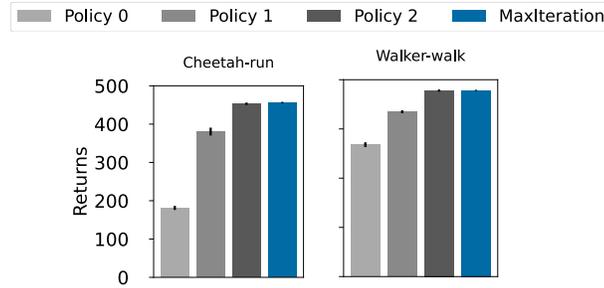



Figure 5: Mean return over 5 seeds of MaxIteration on DM Control tasks [Tunyasuvunakool et al., 2020]. Error-bars correspond to standard error. MaxIteration always selects the best performing constituent policy.

C.4 Computational Resources

Our experiments were conducted using a total of 17 GPUs including both server-grade (e.g., NVIDIA RTX A6000s) and consumer-grade (e.g., NVIDIA RTX 3090) GPUs. Training the constituent policies from offline data takes less than 2 hours. Our MaxIteration algorithm takes about 3 hours to train while the baseline fine-tuning takes around 1 hour. A large chunk of the runtime cost stems from executing the simulator.

17