Appendix for Continual Learning In Environments With Polynomial Mixing Times

Matthew Riemer[*1,2], **Sharath Chandra Raparthy**[*2], **Ignacio Cases**[3], **Gopeshh Subbaraj**[2], **Maximilian Puelma Touzel**[2], **Irina Rish**[2]

A Connections To Literature On Continual And Off-Policy RL

In this section, we begin by highlighting connections to the literature on catastrophic forgetting and task structure in continual RL. Moreover, in the main text we primarily discussed the connection between polynomial mixing times and implications for on-policy model-free and on-policy model-based RL approaches. Here we will discuss connections to off-policy approaches as well.

A.1 Connection to Literature on Continual RL

Catastrophic Forgetting and Experience Replay: In the main text, we began to highlight connections between myopic optimization bias and catastrophic forgetting [McCloskey and Cohen, 1989] in continual RL. Indeed, this often results from myopic updates that do not properly consider the long-term distribution of experiences needed to properly address the stability-plasticity dilemma [Carpenter and Grossberg, 1987] of continual RL as explained in [Khetarpal et al., 2020]. A particularly successful approach in this regard has been those based on experience replay [Lin, 1992] such as recent approaches to continual RL [Isele and Cosgun, 2018, Riemer et al., 2018, Rolnick et al., 2019] or approaches that use a scalable generative form of replay [Riemer et al., 2019]. As highlighted by [Pan et al., 2018], experience replay is closely related to model-based RL approaches. As such, we should only expect experience replay to help us optimize for the infinite horizon objective to the extent that our replay buffer reflects the stationary distribution of the current policy. The likelihood that this would happen by chance is quite small in environments with high mixing times in which the transitions are highly biased towards the transient distribution. As such, off-policy correction or model-based approaches are preferable in the general case. That said, in an environment that passive task switches that are independent of the agent's behavior, maintaining a replay buffer with reservoir sampling [Riemer et al., 2018] will converge in the long-run to the proper steady-state distribution over tasks, only leaving the within task state distribution in need of potential off-policy correction. This insight may shed light on the success of replay based strategies for continual RL, that have been largely tested in settings similar to Example 1 in the main text, particularly considering how they often perform relatively well when it comes to converged final performance.

Definition of Tasks: In Section 2.2 we provide a useful definition of tasks for the purposes of our paper. It is largely inline with the literature that often considers tasks as an unobserved component of the state space. For example, in Bayes-Adaptive MDPs unobserved tasks are sampled every episode from a stationary distribution [Duff, 2002, Ross et al., 2007, Zintgraf et al., 2019]. On the other hand, Hidden-Mode MDPs formalize MDPs where tasks evolve based on a Markov chain that the agent can only passively observe and not influence [Choi et al., 2000, Xie et al., 2020]. Finally, Mixed Observability MDPs (MOMDPs) [Ong et al., 2010] allow for the task space to evolve with transition dynamics that involve the agent's behavior. We can even consider decentralized multi-agent settings with learning agents as a special case of the MOMDP setting [Foerster et al., 2017, Al-Shedivat et al., 2018, Kim et al., 2021]. All of these task specifications fit squarely within our framework, but also

[*]Equal Contribution
[1]IBM Research; [2]Mila, Université de Montréal; [3]Massachusetts Institute of Technology

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

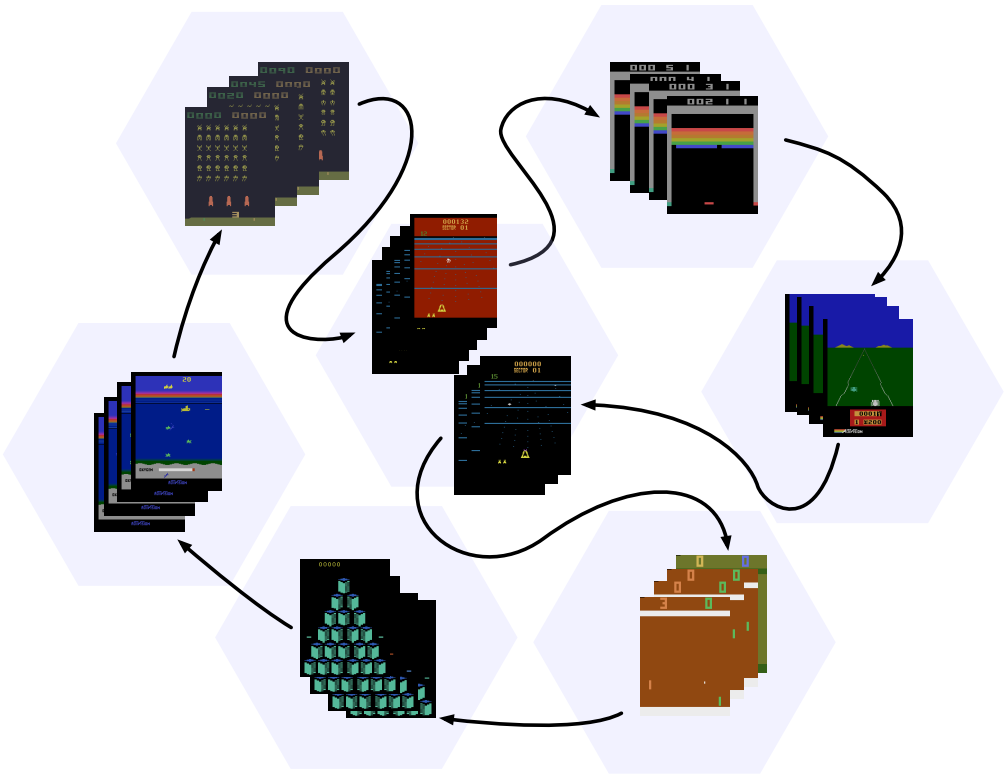

Figure 1: **Task Structure in Atari Experiments:** We depict an arbitrarily selected random path through the seven Atari games considered in our experiments (each task represented by an hexagonal region in the figure). In this specific instantiation, Beam Rider (center) was played twice before each task was visited at least once. Notice how the maximum path length between two tasks must scale at least with the number of tasks as this number is grown.

we do not actually rely on the task being unobserved to highlight the difficulties associated with high mixing times and do not need a formal notion of a task space for the results in our paper to hold.

Tasks That Are Not Explicit: Indeed, similar concepts to tasks may emerge naturally based on the environment structure alone as is common in Factored MDPs [Kearns and Koller, 1999, Boutilier et al., 2000, Strehl et al., 2007, Osband and Van Roy, 2014, Chitnis et al., 2020, Abdulhai et al., 2021]. Moreover, agents may learn to decompose the problem into sub-tasks or options [Sutton et al., 1999] on their own. While some work does exist that models task evolution as a truly non-stationary process [Padakandla et al., 2019, Lecarpentier and Rachelson, 2019, Chandak et al., 2020], as pointed out by [Khetarpal et al., 2020], these models must be very conservative to allow for this non-stationarity and are less likely to be able to exploit regularities between tasks and within task evolution structure as a result.

Illustrating Task Structure in our Experiments: Figure 1 of this appendix represents a simple instantiation of the seven tasks (Atari games) of our experiments. The seven tasks correspond to regions as described in Figure 2 of the main text, and are represented here as frame sequences from the game-play enclosed in hexagonal regions. Each stack of frames has the first frame in the bottom and the last frame up front. The region's boundaries are frames from which the arrows depart; note that these frames connect one region with the *next* (i.e. the arrows always go from the last frame of the current task to the first frame of the next task). Taking them together, the arrows define a specific instantiation of a path through the different regions, which lower bounds the diameter as explained in Figure 2 of the main text.

A.2 Comparison To Off-Policy Learning

Off-Policy Model-Based: In particular, our proposed approaches for tackling myopic bias in model-based RL detailed in Appendix C share similarities with off-policy model-based approaches such as Dyna [Sutton, 1991] and more modern variants for deep RL like MBPO [Janner et al., 2019]. The main advantage of our approach is that it is able to perform policy improvement directly over the limiting distribution and address the pitfall highlighted by Corollary 4 when performing traditional bootstrapping as these approaches do. Addressing this limitation is important in light of results demonstrating that approaches like Dyna benefit greatly from longer rollouts [Holland et al., 2018], which highlights the limitations of traditional bootstrapping. Moreover, we consistently achieve lower regret than Dyna across all of our experiments despite allowing for a very large range of rollouts in our hyperparameter search.

Off-Policy Model-Free: A very popular approach to off-policy model-free RL is to leverage *importance sampling* [Precup, 2000], which attempts to correct data in an off-policy buffer to better mirror the stationary distribution of the current policy. However, traditional importance sampling faces concerns related to both scalability and variance for the long-horizon and continuing problems that we consider in this work [Levine et al., 2020]. More in the spirit of our work are approaches that consider *marginalized importance sampling*, where agents use dynamic programming in order to estimate the ratio between the stationary and behavior distribution from an off-policy buffer. One group of approaches looking to tackle this problem try to estimate this ratio directly leveraging a Bellman equation style update [Hallak and Mannor, 2017, Gelada and Bellemare, 2019, Wen et al., 2020]. These approaches suffer from the same difficulty from Corollary 4 for finding this ratio, making it difficult to successfully apply these approaches in environments with high mixing times. Meanwhile, a second group of approaches attempt to find this ratio by leveraging a learned value function [Nachum et al., 2019a,b, Tang et al., 2019, Nachum and Dai, 2020]. These approaches both clearly do not address Corollary 4, and generally still need expensive optimizations procedures at each step while solving only a regularized (i.e. biased) learning problem. See [Levine et al., 2020] for a comprehensive review of the research highlighted in this paragraph.

B Proposed Algorithms

Estimating The Mixing Time. In this section, we begin by providing pseudo-code for mixing time estimation as outlined in the main text. For the detailed algorithm please refer Algorithm 1. Regarding the hyperparameters, for all our experiments we considered $\mathcal{H} = \tau \times 10^3$. For the experiments reported in Figure 3 and 5, we report the mean and standard deviation across 3 seeds. For Figure 4 experiments, in order to remove the task bias, we randomly shuffle the list of environments we consider and report the mean and standard deviation across 10 seeds.

Mixing Time Independent Algorithms for Efficient Scaling. In the main text we theoretically argued that the polynomial mixing times impact the learning. In this section we will show the empirical evidence on some continual gridworld domains to support those arguments. In order to efficiently learn in the environments with polynomial mixing times, we need to build algorithms which scale in terms of compute independent of the mixing times. While it is challenging to do so in complex domains like Atari, it is an achievable target in tabular domains where we can estimate the steady state distribution of a given policy $\mu^\pi(s)$ in a closed form. More precisely, the steady state equation $\sum_{s\in\mathcal{S}} \mu^\pi(s) \sum_{a\in\mathcal{A}} \pi(a|s) T(s'|s,a) = \mu^\pi(s') \;\; \forall s' \in \mathcal{S}$ can be viewed as a system of equations which can be solved exactly by performing matrix inversion of a $\mathcal{S} \times \mathcal{S}$ matrix. Using the well known linear algebra libraries, we can calculate the steady state distribution in a closed form and perform sampling from this distribution directly for learning. Here we propose two such algorithms which leverage system of equations approach to perform policy improvement with respect to the average reward based on our current model of the environment. We first propose **on-policy ρ learning** which is similar to on-policy q-learning in that optimal actions are considered as we enter each state based on our current approximation of the average reward per step resulting from each action. Next we propose **off-policy ρ learning**, an off-policy version of it where we update the current policy based on the trajectories collected by some behaviour policy. Please refer to Algorithm 2 and Algorithm 3 for more details.

Algorithm 1 ϵ- Return Mixing Time Estimation

Function `MixingTimeEstimation(`env,ϵ, π, $|\hat{\mathcal{S}}|_{max}$`):`

```
// Calculate the asymptotic h-step return of the policy π
s_h ← env.reset()
G(π) ← 0
Initialize H                                   // Horizon length for ρ(π) calculation
for h in 1, 2, ... H do
    a_h ← π(s_h)
    s_{h+1}, r_h ← env.step(a_h)
    G(π) = G(π) + r_h
end
ρ(π) = G(π)/H                                  // Asymptotic h-step return

// Store the reward history for each state
StateRewardHistory ← dict()
s_h ← env.reset()
for h in 1, 2, ... H do
    a_h ← π(s_h)
    s_{h+1}, r_h ← env.step(a_h)
    s_idx ← random.randint(1, h)
    if s_idx < |Ŝ|_max then
        StateRewardHistory[s_idx] ← [ ]
    else
        for s_i in StateRewardHistory do
            StateRewardHistory[s_i].append(r_h)
        end
    end
end
// Calculate the mixing time using ρ(π) and StateRewardHistory
t^π_ret(ε) ← [ ]
for state in StateRewardHistory do
    rewards ← StateRewardHistory[state]
    t_mix ← [ ]
    h ← 0
    G(h, state, π) ← 0
    for r in rewards do
        h ← h + 1
        G(h, state, π) ← G(h, state, π) + r
        ρ(h, state, π) ← G(h, state, π)/h
        if |ρ(h, state, π) − ρ(π)| < ε then
            t_mix.append(h)
        else
            t_mix ← [ ]
        end
    end
    t^π_ret(ε).append(min(t_mix))              // Add min(t_mix) to t^π_ret(ε)
end
return Mean(t^π_ret(ε))
```

Algorithm 2 On-Policy ρ-Learning

Function `OnPolicy`ρ`Learning`(env, ϵ)**:**

Initialize π*,* $\hat{T}$ *and* $\hat{R}$
$s_t \leftarrow env.reset()$
while *not done* **do**
  $p \sim uniform([0, 1])$
  if $p \geq \epsilon$*:* **then**
    for $a \in \mathcal{A}$*:* **do**
      Solve for $\hat{\mu}^{\pi_m(s_t,a,\pi)}$ using $\sum_{s\in\mathcal{S}} \hat{\mu}^{\pi}(s) \sum_{a\in\mathcal{A}} \pi(a|s) T(s'|s, a) = \hat{\mu}^{\pi}(s')$
      Solve for $\hat{\rho}(\pi_m(s_t, a, \pi))$ using $\hat{\rho}(\pi) = \sum_{s\in\mathcal{S}} \hat{\mu}^{\pi}(s) \sum_{a\in\mathcal{A}} \pi(a|s)\hat{R}(s, a)$
      $a_t = \text{argmax}_{a\in\mathcal{A}}\, \hat{\rho}(\pi_m(a, s_t, \pi))$
      Update to greedy policy: $\pi = \pi_m(a_t, s_t, \pi)$
    end
  else
    Random exploration: $a_t \sim uniform(a \in \mathcal{A})$
    $s_{t+1}, r_t \leftarrow env.step(s_t, a_t)$
    Update $\hat{T}$ *on* (s_t, a_t, s_{t+1}) *and* $\hat{R}$ *on* (s_t, a_t, r_t)
    Update to next state: $s_t = s_{t+1}$
  end
end
return $\pi, \hat{T}, \hat{R}$

C Additional Implementation Details and Tabular Experiments

In this section we begin by discussing additional details for about our experiments in the main text and then outline our tabular experiments highlighting how approaches that estimate the steady-state distribution directly can combat myopic bias in the presence of polynomial mixing times.

C.1 Additional Details for Mixing Time Experiments

Compute: Since our mixing time analysis is based on pretrained policies, our experiments are not GPU heavy. However, to get the accurate mixing time estimates we run the experiments for a large number of asymptotic steps $\mathcal{H} = 1000\tau$ (at least on the order of a million steps) and the experiments take long time to run. For small τ values the experiments took significantly less than 24hrs but for larger values of τ the experiments took more than 48hrs. Our experiments were deployed on a cluster of Intel x86 machines requesting 1 GPU (either K40, K80, RTX800 or V100s) and 1 CPU for each experiment. RAM was allocated as appropriate for each experiment with larger values of $\mathcal{H}$ requiring more than 100GB.

For tabular experiments, since all the baselines and our proposed methods are tabular methods, training these do not need huge compute. However, for steady state approximation using exact method we have provided an option of using a pytorch based system of equations solver in the codebase to leverage the fast GPU computations for larger matrices. All the relevant libraries and frameworks used to run the experiments are detailed in the code README file.

C.2 Tabular Experiments

To augment the scalable MDP use cases for deep continual RL highlighted in example 1 in the main text, we consider three grid world based examples with regions of size $d \times d$ in this section to outline some of the key ways that scaling often contributes to polynomial mixing times in practice. We then empirically analyze the scaling behavior of Algorithms 2 and 3 on these environment classes using accumulated lifelong regret per step and contrast this performance against relevant baselines. We perform a grid search over learning rate, exploration parameter ϵ and batchsize B and pick the best

Algorithm 3 Off-Policy ρ-Learning

Function `OffPolicy`ρ`Learning(`env, ϵ, B`):`

Initialize π, $\hat{T}$ *and* $\hat{R}$
$s_t \leftarrow env.reset()$
while *not done* **do**
  $p \sim uniform([0, 1])$
  if *if* $p \geq \epsilon$ **then**
    Sample an action: $a_t \sim \pi(s_t)$
  else
    Random exploration: $a_t \sim uniform(a \in \mathcal{A})$
  end
  $s_{t+1}, r_t \leftarrow env.step(s_t, a_t)$
  Update $\hat{T}$ *on* (s_t, a_t, s_{t+1}) *and* $\hat{R}$ *on* (s_t, a_t, r_t)
  Update to next state: $s_t = s_{t+1}$
  for $i \in [0, ..., B-1]$ **do**
    $s_i \sim uniform(s \in \mathcal{S})$
    for $a \in \mathcal{A}$*:* **do**
      Solve for $\hat{\mu}^{\pi_m(s_t,a,\pi)}$ using $\sum_{s\in\mathcal{S}} \hat{\mu}^{\pi}(s) \sum_{a\in\mathcal{A}} \pi(a|s) T(s'|s,a) = \hat{\mu}^{\pi}(s')$
      Solve for $\hat{\rho}(\pi_m(s_t, a, \pi))$ using $\hat{\rho}(\pi) = \sum_{s\in\mathcal{S}} \hat{\mu}^{\pi}(s) \sum_{a\in\mathcal{A}} \pi(a|s) \hat{R}(s,a)$
      $a_i = \text{argmax}_{a\in\mathcal{A}}\, \hat{\rho}(\pi_m(a, s_i, \pi))$
      Update to greedy policy: $\pi = \pi_m(a_i, s_i, \pi)$
    end
  end
end
return $\pi, \hat{T}, \hat{R}$

Grid Length d	*Steps*	*On-Policy Q-Learning*	*On-Policy ρ-Learning*	*Off-Policy Q-Learning*	*Dyna Q-Learning*	*Model-based n-step TD*	*Off-Policy ρ-Learning*
5	10k	$0.113_{\pm 0.028}$	$0.054_{\pm 0.003}$	$0.100_{\pm 0.058}$	$0.097_{\pm 0.046}$	$0.135_{\pm 0.029}$	$0.041_{\pm 0.003}$
	100k	$0.081_{\pm 0.028}$	$0.048_{\pm 0.004}$	$0.077_{\pm 0.028}$	$0.101_{\pm 0.011}$	$0.117_{\pm 0.024}$	$0.036_{\pm 0.003}$
25	10k	$0.062_{\pm 0.037}$	$0.033_{\pm 0.004}$	$0.058_{\pm 0.039}$	$0.061_{\pm 0.038}$	$0.060_{\pm 0.037}$	$0.057_{\pm 0.015}$
	100k	$0.057_{\pm 0.040}$	$0.016_{\pm 0.002}$	$0.056_{\pm 0.042}$	$0.060_{\pm 0.038}$	$0.059_{\pm 0.037}$	$0.019_{\pm 0.002}$

Table 1: Accumulated lifelong regret per step obtained by an agent in a scalable MDP featuring spatial scaling (Example C.1).

hyperparameters for all models (including the baselines). For each experiment we report the mean and the standard deviation across 10 seeds.

Example C.1 *(Spatial dimensions, d): In this episodic task, the agent is placed at an arbitrary location in a* $d \times d$ *grid world and must reach a goal in an arbitrary location that is fixed across episodes. The agent is only rewarded upon reaching the goal location, which implies that the expected diameter of* π^* *is* $\mathbb{E}[D^{\pi^*}] \in \Omega(d)$*. Since* $|\mathcal{S}| = d^2$ *for this class, we have* $\mathbb{E}[D^{\pi^*}] \in \Omega(|\mathcal{S}|^{\frac{1}{2}})$*.*

Results for $d = 5, 25$ are shown in Table 1. Our proposed algorithms consistently outperform baseline models in terms of lifelong regret.

Example C.2 *(Number of Bottlenecks, N): Consider* N *grid world regions* $\{\mathcal{R}_i\}_{i=1}^N$*, implying* $|\mathcal{S}| = Nd^2$*. Each has an arbitrary starting location when entering from the previous region and an arbitrary goal location serving as a bottleneck transporting the agent to the next region. The agent is only rewarded when it reaches the goal location of its current region, which implies that* $\mathbb{E}[D^{\pi^*}_{\mathcal{R}_i}] \in \Omega(d)$ *for all* i*. We consider three possibilities for how regions are connected. Cycle Transitions: the regions are accessed in a strict order with no repeats. We know that* $\mathbb{E}[D^{\pi^*}] \in \Omega(d \times N)$*, so if* N *is scaled with* d *fixed,* $\mathbb{E}[D^{\pi^*}] \in \Omega(|\mathcal{S}|)$*. Random Transitions: the regions are accessed randomly with repeats. We again know that* $\mathbb{E}[D^{\pi^*}] \in \Omega(d \times N)$*, so again* $\mathbb{E}[D^{\pi^*}] \in \Omega(|\mathcal{S}|)$*. Curricular*

No. of Rooms N	*Task type*	*Steps*	*On-Policy Q-Learning*	*On-Policy ρ-Learning*	*Off-Policy Q-Learning*	*Dyna Q-Learning*	*Model-based n-step TD*	*Off-Policy ρ-Learning*
4	*Random*	10k	$0.367_{\pm 0.138}$	$0.162_{\pm 0.040}$	$0.396_{\pm 0.230}$	$0.300_{\pm 0.124}$	$0.274_{\pm 0.032}$	$0.180_{\pm 0.038}$
		100k	$0.279_{\pm 0.059}$	$0.152_{\pm 0.038}$	$0.242_{\pm 0.090}$	$0.222_{\pm 0.049}$	$0.201_{\pm 0.032}$	$0.153_{\pm 0.040}$
	Cycles	10k	$0.271_{\pm 0.115}$	$0.074_{\pm 0.030}$	$0.287_{\pm 0.161}$	$0.171_{\pm 0.070}$	$0.229_{\pm 0.075}$	$0.099_{\pm 0.033}$
		100k	$0.165_{\pm 0.056}$	$0.063_{\pm 0.029}$	$0.120_{\pm 0.045}$	$0.120_{\pm 0.045}$	$0.130_{\pm 0.042}$	$0.065_{\pm 0.029}$
16	*Random*	10k	$0.365_{\pm 0.230}$	$0.138_{\pm 0.014}$	$0.348_{\pm 0.178}$	$0.410_{\pm 0.087}$	$0.355_{\pm 0.105}$	$0.292_{\pm 0.030}$
		100k	$0.303_{\pm 0.092}$	$0.091_{\pm 0.009}$	$0.321_{\pm 0.095}$	$0.187_{\pm 0.017}$	$0.334_{\pm 0.030}$	$0.106_{\pm 0.010}$
	Cycles	10k	$0.338_{\pm 0.162}$	$0.100_{\pm 0.026}$	$0.303_{\pm 0.152}$	$0.364_{\pm 0.084}$	$0.412_{\pm 0.114}$	$0.287_{\pm 0.041}$
		100k	$0.243_{\pm 0.067}$	$0.062_{\pm 0.017}$	$0.363_{\pm 0.122}$	$0.144_{\pm 0.028}$	$0.181_{\pm 0.04}$	$0.083_{\pm 0.017}$
2	*Curricular*	10k	$0.452_{\pm 0.128}$	$0.343_{\pm 0.071}$	$0.379_{\pm 0.0.095}$	$0.400_{\pm 0.086}$	$0.408_{\pm 0.096}$	$0.354_{\pm 0.080}$
		100k	$0.424_{\pm 0.109}$	$0.340_{\pm 0.067}$	$0.362_{\pm 0.098}$	$0.366_{\pm 0.092}$	$0.384_{\pm 0.087}$	$0.340_{\pm 0.070}$
3	*Curricular*	10k	$0.359_{\pm 0.157}$	$0.281_{\pm 0.043}$	$0.389_{\pm 0.234}$	$0.322_{\pm 0.091}$	$0.331_{\pm 0.118}$	$0.260_{\pm 0.055}$
		100k	$0.306_{\pm 0.097}$	$0.259_{\pm 0.073}$	$0.306_{\pm 0.097}$	$0.300_{\pm 0.076}$	$0.285_{\pm 0.076}$	$0.250_{\pm 0.063}$
4	*Curricular*	10k	$0.283_{\pm 0.059}$	$0.180_{\pm 0.056}$	$0.311_{\pm 0.094}$	$0.246_{\pm 0.042}$	$0.286_{\pm 0.074}$	$0.195_{\pm 0.071}$
		100k	$0.245_{\pm 0.069}$	$0.161_{\pm 0.053}$	$0.207_{\pm 0.037}$	$0.210_{\pm 0.045}$	$0.209_{\pm 0.055}$	$0.183_{\pm 0.076}$

Table 2: Accumulated lifelong regret per step obtained by an agent in a scalable MDP featuring bottleneck scaling (Example C.2). The values shown are for the three room transition variants across different N values with each room of size $d = 5$.

Exponent x	*Steps*	*On-Policy Q-Learning*	*On-Policy ρ-Learning*	*Off-Policy Q-Learning*	*Dyna Q-Learning*	*Model-based n-step TD*	*Off-Policy ρ-Learning*
2	10k	$0.245_{\pm 0.016}$	$0.125_{\pm 0.008}$	$0.286_{\pm 0.024}$	$0.240_{\pm 0.018}$	$0.279_{\pm 0.025}$	$0.216_{\pm 0.010}$
	100k	$0.243_{\pm 0.013}$	$0.062_{\pm 0.002}$	$0.233_{\pm 0.020}$	$0.124_{\pm 0.012}$	$0.201_{\pm 0.012}$	$0.070_{\pm 0.002}$
3	10k	$0.233_{\pm 0.024}$	$0.089_{\pm 0.005}$	$0.253_{\pm 0.024}$	$0.262_{\pm 0.012}$	$0.262_{\pm 0.012}$	$0.252_{\pm 0.016}$
	100k	$0.197_{\pm 0.019}$	$0.049_{\pm 0.003}$	$0.194_{\pm 0.022}$	$0.121_{\pm 0.011}$	$0.168_{\pm 0.013}$	$0.072_{\pm 0.003}$
4	10k	$0.220_{\pm 0.036}$	$0.066_{\pm 0.007}$	$0.243_{\pm 0.046}$	$0.269_{\pm 0.085}$	$0.243_{\pm 0.051}$	$0.236_{\pm 0.034}$
	100k	$0.185_{\pm 0.014}$	$0.047_{\pm 0.003}$	$0.168_{\pm 0.014}$	$0.140_{\pm 0.012}$	$0.163_{\pm 0.015}$	$0.089_{\pm 0.007}$

Table 3: Accumulated lifelong regret per step obtained by an agent in a scalable MDP featuring cycle length scaling (Example C.3). The values shown are for the *cyclic* room transitions with $N = 16$ rooms.

Transitions: the regions are accessed in a curricular fashion i.e. $\mathcal{R}_1, \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$, *etc. We know* $\mathbb{E}[D^{\pi^*}] \in \Omega(d \times N!)$*, so if* N *is scaled and* d *is fixed,* $\mathbb{E}[D^{\pi^*}] \in \Omega(|\mathcal{S}|^{N-1})$.

Results for $N = 4, 16$ and $d = 5$ are shown in Table 2. Our models consistently outperform baselines for both cyclic and random transitions regardless of the number of rooms. The curricular transition case is challenging since the diameter scales exponentially with N, making the diameter substantially higher. The corresponding convergence rates of all methods (Table 2) are lower as expected by regret bounds [Jaksch et al., 2010]. Nevertheless, our proposed models still outperform the baselines.

Example C.3 *(Cycle length,* τ*):* N *grid world regions with cyclic transitions for which the agent is expected to transition between regions after every* τ *environment steps. We use cyclic transitions to highlight similarities to typical settings in continual RL [Khetarpal et al., 2020]. Here, room transitions are passive: the agents' current policy has no direct effect on the room transitions which purely depend on* τ*.* $\tau \geq 2d$ *as otherwise some regions would be impossible to solve, so we can assume a form* $\tau = cd^x \;\; \forall c \geq 2, x \geq 1$*. This implies that as we scale* x *keeping* c, d*, and* N *constant,* $\mathbb{E}[D^{\pi^*}] \in \Omega(|\mathcal{S}|^{x/2})$.

Note that this example bears strong similarity to Example 1 discussed in the main text and can be seen as the grid world analog of that setting. Results for $x = 2, 3$ and 4 keeping $d = 5$ and $N = 16$ fixed are shown in Table 3. As expected, our proposed methods achieve better sample efficiency.

D Detailed Proofs For Key Results

In this section, we provide detailed proofs for all of the propositions and theorems in our paper following the order that they are presented in the main text. For each proof we first remind readers of the main result and provide a proof sketch before detailing how each step is achieved.

D.0 Formal Description of MDP Scaling

In this section, we formalize a scalable family of MDPs $\mathbb{C}_\sigma$, using a scaling function, σ. We consider MDPs with state space $\mathcal{S}$ parametrized by an n-dimensional vector $\boldsymbol{q} \in \mathbb{R}^n$. In general, $\boldsymbol{q}$ can contain spatial components determining spatial properties of $\mathcal{S}$ (e.g. spatial dimensions of a grid world), as

well as non-spatial components that determine qualitative features (e.g. number of 'cherry' states in a rewarded sequence of target states in a grid world). As a continuous parametrization of the MDP, $\boldsymbol{q}$ serves as a useful target for formalizing scaling. We formulate scaling using the following definition:

Definition: A *scaling function* is a continuous deformation $\sigma : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ parametrized by a scalar $\nu \in \mathbb{R}$ that takes any $\boldsymbol{q}_0$ to $\boldsymbol{q}_\nu = \sigma(\boldsymbol{q}_0, \nu)$, with $\sigma(\cdot, 0)$ as the identity map.

We denote by Σ the set of all scaling functions on $\boldsymbol{q}$. Thus, each $\sigma \in \Sigma$ induces $\mathbb{C}_\sigma = \{\mathcal{M}_\nu\}$, a ν-parametrized abstract class of MDPs along a smooth path in q-space. We consider scaling functions that scale up the MDP:

Definition: An *expansive scaling function* is a scaling function for which $\partial q_{\nu,i}/\partial\nu \geq 0$ for all $i = 1, \ldots, n$, and $\partial q_{\nu,i}/\partial\nu > 0$ for at least one i.

Thus, the MDP will never decrease in size along a path in $\boldsymbol{q}$-space taken through an expansive scaling function. An important example of expansive scaling functions is:

Definition: A *proportional scaling function* is an expansive scaling function taking the form $\sigma(\boldsymbol{q}, \nu) = \boldsymbol{q} + \nu\Delta\boldsymbol{q}$, where $\Delta\boldsymbol{q} \in \mathbb{R}^n$ and Δq_i is the linear rate of grow of $q_{\nu,i}$ with ν.

We focus on proportional scaling functions in our definition of scalable MDPs stated again in D.3.

D.1 Proof of Proposition 1

Proposition 1 *If all MDPs $\mathcal{M}$ within the subclass of MDPs $\mathbb{C}$ have $t_{\text{ret}}^{\pi^*}$, $t_{\text{ces}}^{\pi^*}$, $t_{\text{mix}}^{\pi^*}$, D^{π^*}, or $D^* \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$ we can say that $\mathbb{C}$ has a polynomial mixing time.*

Proposition 1 is thus reliant on the following definition of a *polynomial mixing time*:

Definition 2: *A set or family of MDPs $\mathbb{C}$ has a **polynomial mixing time** if the environment mixing dynamics contributes a $\Omega(|\mathcal{S}|^k)$ multiplicative increase for some $k > 0$ to the intrinsic lower bound on regret as $|\mathcal{S}| \to \infty \ \ \forall \mathcal{M} \in \mathbb{C}$.*

Proof Sketch: It holds directly from Jaksch et al. [2010] Theorem 5 that following Assumption 1 from the main text yields a lower bound on regret for RL algorithms $\text{Regret}(H) \in \Omega(\sqrt{D^*|\mathcal{S}||\mathcal{A}|H})$, which implies that if $D^* \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$, there must be a *polynomial mixing time*. We then begin by noting how this regret bound holds equally for D^{π^*}. We proceed to demonstrate that if $t_{\text{mix}}^{\pi^*}$ is polynomial in $|\mathcal{S}|$, the same must be true for D^{π^*}. We then establish the relationship between $t_{\text{ret}}^{\pi^*}$ and $t_{\text{mix}}^{\pi^*}$ and go on to establish the relationship between $t_{\text{ces}}^{\pi^*}$ and $t_{\text{mix}}^{\pi^*}$. As such, it is demonstrated that if any of these metrics have a polynomial dependence, the regret bound must as well.

Analysis for D^{π^*}: As discussed extensively in [Osband and Van Roy, 2016], the common approach to proving regret bounds in RL and bandit settings it to develop a counter-example for which you can demonstrate that it is impossible for an algorithm to get below a certain regret as a function of the problem parameters. This is the approach taken by Jaksch et al. [2010] to establish the $\text{Regret}(H) \in \Omega(\sqrt{D^*|\mathcal{S}||\mathcal{A}|H})$ bound. However, a careful analysis of their proof reveals that for the problem they consider $D^* = D^{\pi^*}$. As such the proof of Theorem 5 in [Jaksch et al., 2010] can equally be used to establish that $\text{Regret}(H) \in \Omega(\sqrt{D^{\pi^*}|\mathcal{S}||\mathcal{A}|H})$. Note that since by definition $\Omega(\sqrt{D^{\pi^*}|\mathcal{S}||\mathcal{A}|H}) \subseteq \Omega(\sqrt{D^*|\mathcal{S}||\mathcal{A}|H})$, the more general result is for D^*. Moreover, if this were not the case it would invalidate the lower bound presented by [Kearns and Singh, 2002] in terms of $t_{\text{ret}}^{\pi^*}$. As such, it is clear that we cannot simply ignore the size of the problem from the perspective of the optimal policy π^* when conducting regret analysis.

Analysis for $t_{\text{mix}}^{\pi^*}$: It is well known that in bounded degree transitive graphs the mixing time $t_{\text{mix}} \in O(D^3)$ where D is the graph diameter [Levin and Peres, 2017]. Note, however, that it is widely conjectured to be at most $t_{\text{mix}} \in O(D^2)$ [Levin and Peres, 2017]. Taking the more general case, this implies then that for any policy π and MDP $\mathcal{M}$, $D^\pi \in \Omega((t_{\text{mix}}^\pi)^{1/3})$. Therefore, if $t_{\text{mix}}^{\pi^*} \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$, then the regret must be in $\text{Regret}(H) \in \Omega(\sqrt{|\mathcal{S}|^{k/3}|\mathcal{S}||\mathcal{A}|H})$. Therefore, Definition 2 is satisfied.

Analysis for $t_{\text{ret}}^{\pi^*}$: It was established in Lemma 1 of [Kearns and Singh, 2002] that $t_{\text{mix}}^\pi \in \Omega(t_{\text{ret}}^\pi)$ for any policy π. So this then implies that $t_{\text{mix}}^{\pi^*} \in \Omega(t_{\text{ret}}^{\pi^*})$. Therefore, we can follow the result from the

last paragraph to see that if $t_{\text{ret}}^{\pi^*} \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$, it must also be true that Definition 2 is satisfied.

Analysis for $t_{\text{ces}}^{\pi^*}$: It has been established that $7t_{\text{mix}} \geq t_{\text{ces}}$ for any finite chain [Levin and Peres, 2017]. Therefor, $t_{\text{mix}}^{\pi^*} \in \Omega(t_{\text{ces}}^{\pi^*})$. So by extension of our results presented above, if $t_{\text{ces}}^{\pi^*} \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$, Definition 2 must hold.

D.2 Proof of Proposition 2

Proposition 2: *Any scalable MDPs $\mathbb{C}_\sigma$ exhibits a polynomial mixing time if there exists a scalable region $\mathcal{R}_\nu$ such that $\mathbb{E}_{\mu^{\pi^*}}[t_{\mathcal{R}_\nu}^{\pi^*}] \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$.*

Proof Sketch: We first present the bottleneck ratio of a region. We then demonstrate the relationship connecting the bottleneck ratio to the residence time. We conclude by establishing the connection between the bottleneck ratio and the mixing time of the optimal policy. This then allows us to connect the residence time of the optimal policy to its mixing time, which allows us to establish a polynomial mixing time leveraging the result from Proposition 1.

We begin by defining an edge measure (or ergodic flow) as $\xi^\pi(s'|s) := \mu^\pi(s)T^\pi(s'|s)$ [Levin and Peres, 2017]. For reasoning about regions of state space $\mathcal{R}$, we can further define an edge measure as $\xi^\pi(\mathcal{S} \setminus \mathcal{R}|\mathcal{R}) = \sum_{s' \in \mathcal{S}\setminus\mathcal{R}} \sum_{s \in \mathcal{R}} \xi^\pi(s'|s)$ and steady-state probability $\mu^\pi(\mathcal{R}) = \sum_{s\in\mathcal{R}} \mu^\pi(s)$. This can then be used to define the bottleneck ratio of the set $\mathcal{R}$ [Levin and Peres, 2017]:

$$\mho^\pi(\mathcal{R}) := \frac{\xi^\pi(\mathcal{S} \setminus \mathcal{R}|\mathcal{R})}{\mu^\pi(\mathcal{R})} = \frac{1}{\mathbb{E}_{\mu^\pi}[t_{\mathcal{R}}^\pi]} \tag{1}$$

With the last equality we note the relationship between the bottleneck ratio of a region $\mathcal{R}$ and the *expected residence time* $\mathbb{E}_{\mu^\pi}[t_{\mathcal{R}}^\pi]$, which is the expected probability of being in $\mathcal{R}$ divided by the expected probability of exiting $\mathcal{R}$. We can finally now define the bottleneck ratio of the entire Markov chain $T^\pi(s'|s)$ (also called the conductance or Cheeger constant) as $\mho_*^\pi := \min_{\mathcal{R}\subset\mathcal{S}:\mu^\pi(\mathcal{R})\leq 1/2} \mho^\pi(\mathcal{R})$ [Levin and Peres, 2017]. Importantly the bottleneck ratio can be used to both upper bound and lower bound on the mixing time [Levin and Peres, 2017]. For our purposes, we will primarily utilize the fact that $t_{\text{mix}}^\pi(1/4) \geq 1/4\mho_*^\pi$ as stated in Theorem 7.4 of [Levin and Peres, 2017]. We can then consider the context of π^*:

$$\begin{aligned} t_{\text{mix}}^{\pi^*}(1/4) \geq \frac{1}{4\mho_*^{\pi^*}} &= \max_{\mathcal{R}\subset\mathcal{S}:\mu^{\pi^*}(\mathcal{R})\leq 1/2} \frac{1}{4\mho^{\pi^*}(\mathcal{R})} \\ &= \max_{\mathcal{R}\subset\mathcal{S}:\mu^{\pi^*}(\mathcal{R})\leq 1/2} \frac{\mathbb{E}_{\mu^{\pi^*}}[t_{\mathcal{R}}^{\pi^*}]}{4} \end{aligned} \tag{2}$$

This in turn implies the following result for the conventional mixing time $t_{\text{mix}}^{\pi^*}(1/4)$:

$$t_{\text{mix}}^{\pi^*}(1/4) \in \Omega(\mathbb{E}_{\mu^{\pi^*}}[t_{\mathcal{R}}^{\pi^*}]) \;\; \forall \mathcal{R} \subset \mathcal{S} : \mu^{\pi^*}(\mathcal{R}) \leq 1/2 \tag{3}$$

This result is nearly what we are looking for. However, we would like to express the mixing time at an arbitrary precision ϵ in terms of the residence time rather than only considering $\epsilon = 1/4$. To achieve this we note that the proof of Theorem 7.4 in [Levin and Peres, 2017] can easily be extended for arbitrary ϵ with $\epsilon = 1/4$ only being used as a convention in the literature. Taking the results presented after equation 7.10 in [Levin and Peres, 2017] and rearranging them with our notation, we find that:

$$t_{\text{mix}}^{\pi^*}(\epsilon) \geq \frac{1 - \epsilon - \mu^{\pi^*}(\mathcal{R})}{\mho_*^{\pi^*}} \tag{4}$$

As a result as long as we restrict $\mu^{\pi^*}(\mathcal{R}) \leq 1 - \epsilon - \delta$ for some $0 < \delta < 1 - \epsilon$ we can conclude that $t^{\pi^*}_{\text{mix}}(\epsilon) \geq \delta t^{\pi^*}_{\mathcal{R}}$. This implies that we can state the following asymptotic result as long as the inequality on the density is strict, leading to some effective positive δ:

$$t^{\pi^*}_{\text{mix}}(\epsilon) \in \Omega(\mathbb{E}_{\mu^{\pi^*}}[t^{\pi^*}_{\mathcal{R}}]) \;\; \forall \mathcal{R} \subset \mathcal{S} : \mu^{\pi^*}(\mathcal{R}) < 1 - \epsilon \tag{5}$$

Therefore, if any scalable region $\mathcal{R}_\nu$ exists where $\mu^{\pi^*}(\mathcal{R}_\nu) < 1 - \epsilon$ and $\mathbb{E}_{\mu^{\pi^*}}[t^{\pi^*}_{\mathcal{R}_\nu}] \in \Omega(|\mathcal{S}|^k)$ for some $k > 0$, we also know that $t^{\pi^*}_{\text{mix}}(\epsilon) \in \Omega(|\mathcal{S}|^k)$ and thus a polynomial mixing time is ensured following Proposition 1. Moreover, we can finalize our proof of Proposition 2 by noting that any arbitrarily large scalable region of density greater than the bound i.e. $\mu^{\pi^*}(\mathcal{R}_\nu) \geq 1 - \epsilon$ that scales polynomially must also contain a scalable sub-region $\mathcal{R'}_\nu \subset \mathcal{R}_\nu$ of arbitrarily small density $\mu^{\pi^*}(\mathcal{R'}_\nu) < 1 - \epsilon$ that scales at least at the same rate. As such, there is no need to provide an upper bound for the steady-state region probability density in Proposition 2.

D.3 Proof of Theorem 1

Theorem 1: *(Mixing Time Scaling): Any scalable MDP $\mathbb{C}_\sigma$ has a polynomial mixing time.*

Theorem 1 uses the following definitions of *scalable MDP* and *polynomial mixing time*.

Definition 1: *A **scalable MDP** is a family of MDPs $\mathbb{C}_\sigma = \{\mathcal{M}_\nu\}$ arising from a proportional scaling function σ satisfying the property that there exists an initial scalable region $\mathcal{R}_0$ with finite interior, $\mu^{\pi^*}(\partial\mathcal{R}_0) < \mu^{\pi^*}(\mathcal{R}_0)$, that scales so that $\mu^{\pi^*}(\partial\mathcal{R}_\nu) < \mu^{\pi^*}(\mathcal{R}_\nu)$ as $\nu \to \infty$ and thus $|\mathcal{S}| \to \infty$.*

Definition 2: *A set or family of MDPs $\mathbb{C}$ has a **polynomial mixing time** if the environment mixing dynamics contributes a $\Omega(|\mathcal{S}|^k)$ multiplicative increase for some $k > 0$ to the intrinsic lower bound on regret as $|\mathcal{S}| \to \infty \;\; \forall \mathcal{M} \in \mathbb{C}$.*

Proof Sketch: We begin by further analyzing the bottleneck ratio and residence time introduced in Proposition 2 by connecting it to the relative densities of a region and its boundary. We then consider how, under the conditions we consider, a proportionally scaled region must grow faster than its boundary and the region to boundary ratio is lower bounded by the respective increase in their sizes as ν increases. Finally, we establish that the scaling of some regions of any scalable MDP undergoing proportional scaling is polynomially larger than the scaling of its boundary and prove polynomial mixing through the use of Proposition 1.

We begin by further analyzing the ergodic flow of any policy π in region $\mathcal{R}$ and boundary $\partial\mathcal{R}$:

$$\begin{aligned} \xi^\pi(\mathcal{S} \setminus \mathcal{R}|\mathcal{R}) &= \sum_{s' \in \mathcal{S}\setminus\mathcal{R}} \sum_{s \in \mathcal{R}} \mu^\pi(s) T^\pi(s'|s) \\ &= \sum_{s' \in \mathcal{S}\setminus\mathcal{R}} \sum_{s \in \partial\mathcal{R}} \mu^\pi(s) T^\pi(s'|s) \leq \mu^\pi(\partial\mathcal{R}) \end{aligned} \tag{6}$$

As such, we can use this result to provide a lower bound for the expected residence time:

$$\mathbb{E}_{\mu^\pi}[t^\pi_{\mathcal{R}}] = \frac{\mu^\pi(\mathcal{R})}{\xi^\pi(\mathcal{S} \setminus \mathcal{R}|\mathcal{R})} \geq \frac{\mu^\pi(\mathcal{R})}{\mu^\pi(\partial\mathcal{R})} \tag{7}$$

This in turn implies the following relation to the mixing time by subbing in equation 5:

$$t^{\pi^*}_{\text{mix}}(\epsilon) \in \Omega(\mathbb{E}_{\mu^{\pi^*}}[t^{\pi^*}_{\mathcal{R}}]) \in \Omega\left(\frac{\mu^{\pi^*}(\mathcal{R})}{\mu^{\pi^*}(\partial\mathcal{R})}\right) \;\; \forall \mathcal{R} \subset \mathcal{S} : \mu^{\pi^*}(\mathcal{R}) < 1 - \epsilon \tag{8}$$

From this analysis it is clear that we can characterize polynomial mixing times by understanding how the probability mass on any scalable region and the mass on its boundary scale with the state space. Formally, for some scaling function σ with scaling parameter ν giving a state space $\mathcal{S}_\nu$, we would like to understand how the mass on a scalable region $\mathcal{R}_\nu$ and its boundary, $\partial\mathcal{R}_\nu$, $\mu^{\pi^*}(\mathcal{R}_\nu)$ and

$\mu^{\pi^*}(\partial\mathcal{R}_\nu)$, respectively, scale with $|\mathcal{S}_\nu|$. Without loss of generality, we set a reference MDP at $\nu = 0$ and denote some region in it as $\mathcal{R}_0$. We suppress the dependence of π^* and μ^{π^*} on ν in our notation for clarity. We now derive the main result. By the definition of a scalable MDP, the bulk-to-boundary mass ratio of a region can not decrease as ν scales up the state space. Thus,

$$\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)} \geq \frac{\mu^{\pi^*}(\mathcal{R}_0)}{\mu^{\pi^*}(\partial\mathcal{R}_0)} \,. \tag{9}$$

Since mass on a scaled region and its boundary cannot increase with ν, $\mu^{\pi^*}(\mathcal{R}_\nu) \leq \mu^{\pi^*}(\mathcal{R}_0)$ and $\mu^{\pi^*}(\partial\mathcal{R}_\nu) \leq \mu^{\pi^*}(\partial\mathcal{R}_0)$. Thus,

$$\begin{aligned}
\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)} &\geq \frac{\mu^{\pi^*}(\mathcal{R}_0)}{\mu^{\pi^*}(\partial\mathcal{R}_0)} \cdot \frac{1 - \frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\mathcal{R}_0)}}{1 - \frac{\mu^{\pi^*}(\partial\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_0)}} \\
\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)} &\geq \frac{\mu^{\pi^*}(\mathcal{R}_\nu) - \mu^{\pi^*}(\mathcal{R}_0)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu) - \mu^{\pi^*}(\partial\mathcal{R}_0)} \,.
\end{aligned} \tag{10}$$

To demonstrate the scaling of the right hand side, we can approximate the numerator and denominator. For sufficiently large $\mathcal{R}$, the variation of the size of and mass on the region and its boundary vary in an approximately continuous way with ν so we can employ the inverse function theorem to expand the mass on a region in its size, $|\mathcal{R}_\nu|$, and the same with its boundary, $|\partial\mathcal{R}_\nu|$. To first order around $\nu = 0$, we have

$$\mu^{\pi^*}(\mathcal{R}_\nu) \approx \mu^{\pi^*}(\mathcal{R}_0) + \frac{\partial\mu^{\pi^*}}{\partial|\mathcal{R}_\nu|}\bigg|_{\nu=0} \Delta|\mathcal{R}| \,, \tag{11}$$

$$\mu^{\pi^*}(\partial\mathcal{R}_\nu) \approx \mu^{\pi^*}(\partial\mathcal{R}_0) + \frac{\partial\mu^{\pi^*}}{\partial|\partial\mathcal{R}_\nu|}\bigg|_{\nu=0} \Delta|\partial\mathcal{R}| \,, \tag{12}$$

for deviations, $\Delta|\mathcal{R}| := |\mathcal{R}_\nu| - |\mathcal{R}_0|$ and $\Delta|\partial\mathcal{R}| := |\partial\mathcal{R}_\nu| - |\partial\mathcal{R}_0|$ (note that the notation $\partial/\partial|\partial\mathcal{R}|$ is the partial derivative with respect to the size of the region's boundary, $\partial\mathcal{R}$). Then, arranging the terms:

$$\frac{\mu^{\pi^*}(\mathcal{R}_\nu) - \mu^{\pi^*}(\mathcal{R}_0)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu) - \mu^{\pi^*}(\partial\mathcal{R}_0)} \approx \frac{\frac{\partial\mu^{\pi^*}}{\partial|\mathcal{R}_\nu|}\big|_{\nu=0}\Delta|\mathcal{R}|}{\frac{\partial\mu^{\pi^*}}{\partial|\partial\mathcal{R}_\nu|}\big|_{\nu=0}\Delta|\partial\mathcal{R}|} \,. \tag{13}$$

Subbing into equation 10 above we get:

$$\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)} \geq \frac{\frac{\partial\mu^{\pi^*}}{\partial|\mathcal{R}_\nu|}\big|_{\nu=0}\Delta|\mathcal{R}|}{\frac{\partial\mu^{\pi^*}}{\partial|\partial\mathcal{R}_\nu|}\big|_{\nu=0}\Delta|\partial\mathcal{R}|} \,. \tag{14}$$

Noting that the derivatives are of equal sign (less than or equal to zero) following Definition 1, we can now state the following scaling dependence:

$$\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)} \in \Omega\left(\frac{\Delta|\mathcal{R}|}{\Delta|\partial\mathcal{R}|}\right) \;\; \forall\mathcal{R}_\nu \subset \mathcal{S}_\nu \,. \tag{15}$$

To further analyze this situation, let's consider the case where the scalable MDP is proportionally scaled among $n' \leq n$ of its n continuous parameters and $n'' \leq n'$ of these parameters result in scaling of a particular region $\mathcal{R}_\nu$. To provide a scaling dependence to $\frac{\Delta|\mathcal{R}|}{\Delta|\partial\mathcal{R}|}$, we utilize the relationship between the scaling of $|\mathcal{R}_\nu|$ and the scaling of the $n'' \leq n'$ intrinsic dimensions controlling its size. As in the main text, we consider the important example of proportional scaling such that the scaling of these n'' dimensions inherits the scaling behaviour of an n''-dimensional hyper-sphere with radius ν. As such the state space scaling follows $|\mathcal{R}_\nu| \in \Theta(\nu^{n''})$. Moreover, the volume-to-surface area ratio follows $\Delta|\mathcal{R}|/\Delta|\partial\mathcal{R}| \in \Theta(\nu)$ in the case of the proportional scaling of Definition 1.[2] This then implies that $\Delta|\mathcal{R}|/\Delta|\partial\mathcal{R}| \in \Theta(|\mathcal{S}_\nu|^{1/n'})$, since $|\mathcal{S}_\nu| \in \Theta(\nu^{n'})$ where the initial size $|\mathcal{S}_0|$ can be omitted from asymptotic notation. Finally, we can substitute this expression into equation 15:

$$\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)} \in \Omega\left(\frac{\Delta|\mathcal{R}|}{\Delta|\partial\mathcal{R}|}\right) \in \Omega\left(|\mathcal{S}_\nu|^{1/n'}\right) \in \Omega\left(|\mathcal{S}_\nu|^{1/n}\right) \;\; \forall\mathcal{R}_\nu \subset \mathcal{S}_\nu \tag{16}$$

[2]This can be seen as using an n-dimensional generalization of Galileo's famous square-cube law.

We can then plug this result into equation 17, yielding a lower bound for $t^{\pi^*}_{\text{mix}}(\epsilon)$:

$$t^{\pi^*}_{\text{mix}}(\epsilon) \in \Omega\left(\frac{\mu^{\pi^*}(\mathcal{R}_\nu)}{\mu^{\pi^*}(\partial\mathcal{R}_\nu)}\right) \in \Omega\left(\frac{\Delta|\mathcal{R}|}{\Delta|\partial\mathcal{R}|}\right) \in \Omega\left(|\mathcal{S}_\nu|^{1/n}\right) \;\; \forall \mathcal{R}_\nu \subset \mathcal{S}_\nu : \mu^{\pi^*}(\mathcal{R}_\nu) < 1-\epsilon \quad (17)$$

As we noted in our proof of Proposition 2, any arbitrarily large scalable region of density greater than the bound i.e. $\mu^{\pi^*}(\mathcal{R}_\nu) \geq 1-\epsilon$ that scales polynomially must also contain a scalable sub-region $\mathcal{R'}_\nu \subset \mathcal{R}_\nu$ of arbitrarily small density $\mu^{\pi^*}(\mathcal{R'}_\nu) < 1-\epsilon$ that scales at least at the same rate. As such, there is no need to provide an upper bound for the steady-state region probability density in Theorem 1. If a scalable MDP is defined, it must be that such a region exists. As we know that $n \geq n' \geq n'' \geq 1$, there must exist a scalable region $\mathcal{R}_\nu$ that has a residence time scaling with $\Omega(|\mathcal{S}_\nu|^{1/n})$ that meets our definition of a polynomial with exponent $k > 0$ for any scalable MDP with a finite description in terms of n continuous variables. The existence of a polynomial mixing time is then proven using Proposition 1.

E Broader Impact Statement

Our work focuses on highlighting the existence of polynomial mixing times as MDPs are scaled up and the resulting myopic bias for common approaches to RL in this setting. Approaches that suffer from myopic bias during learning may suffer from various sources of optimization instability that could potentially be harmful to society if experienced by agents deployed to manage high impact services and applications. For example, a myopic agent may only optimize its behavior for the short term future, even if this will result in catastrophic effects for society in the long-term. As such, addressing myopic bias resulting from polynomial mixing times is a key step towards developing reliable and trustworthy RL agents. Our work takes a first step towards this goal by outlining the foundational theory underlying the issue and we hope that future work can use these insights towards developing agents that can robustly learn to tackle large-scale real-world problems.

References

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation, 24:109–165*, 1989.

Gail A Carpenter and Stephen Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer vision, graphics, and image processing*, 37(1): 54–115, 1987.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *arXiv preprint arXiv:2012.13490*, 2020.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32:350–360, 2019.

Matthew Riemer, Tim Klinger, Djallel Bouneffouf, and Michele Franceschini. Scalable recollections for continual lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1352–1359, 2019.

Yangchen Pan, Muhammad Zaheer, Adam White, Andrew Patterson, and Martha White. Organizing experience: a deeper look at replay mechanisms for sample-based planning in continuous state domains. *arXiv preprint arXiv:1806.04624*, 2018.

Michael O'Gordon Duff. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. PhD thesis, University of Massachusetts at Amherst, 2002.

Stephane Ross, Brahim Chaib-draa, and Joelle Pineau. Bayes-adaptive pomdps. In *NIPS*, pages 1225–1232, 2007.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

Samuel PM Choi, Dit-Yan Yeung, and Nevin L Zhang. Hidden-mode markov decision processes for nonstationary sequential decision making. In *Sequence Learning*, pages 264–287. Springer, 2000.

Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst lifelong non-stationarity, 2020.

Sylvie CW Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *The International Journal of Robotics Research*, 29(8):1053–1068, 2010.

Jakob N Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*, 2017.

Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. *ICLR*, 2018.

Dong Ki Kim, Miao Liu, Matthew D Riemer, Chuangchuang Sun, Marwa Abdulhai, Golnaz Habibi, Sebastian Lopez-Cot, Gerald Tesauro, and Jonathan How. A policy gradient algorithm for learning to learn in multiagent reinforcement learning. In *International Conference on Machine Learning*, pages 5541–5550. PMLR, 2021.

Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *IJCAI*, volume 16, pages 740–747, 1999.

Craig Boutilier, Richard Dearden, and Moisés Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial intelligence*, 121(1-2):49–107, 2000.

Alexander L Strehl, Carlos Diuk, and Michael L Littman. Efficient structure learning in factored-state mdps. In *AAAI*, volume 7, pages 645–650, 2007.

Ian Osband and Benjamin Van Roy. Near-optimal reinforcement learning in factored mdps. *Advances in Neural Information Processing Systems*, 27:604–612, 2014.

Rohan Chitnis, Tom Silver, Beomjoon Kim, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Camps: Learning context-specific abstractions for efficient planning in factored mdps. *arXiv preprint arXiv:2007.13202*, 2020.

Marwa Abdulhai, Dong-Ki Kim, Matthew Riemer, Miao Liu, Gerald Tesauro, and Jonathan P How. Context-specific representation abstraction for deep option learning. *arXiv preprint arXiv:2109.09876*, 2021.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Sindhu Padakandla, Shalabh Bhatnagar, et al. Reinforcement learning in non-stationary environments. *arXiv preprint arXiv:1905.03970*, 2019.

Erwan Lecarpentier and Emmanuel Rachelson. Non-stationary markov decision processes, a worst-case approach using model-based reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 7214–7223, 2019.

Yash Chandak, Georgios Theocharous, Shiv Shankar, Sridhar Mahadevan, Martha White, and Philip S Thomas. Optimizing for the future in non-stationary mdps. *arXiv preprint arXiv:2005.08158*, 2020.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*, 2019.

G Zacharias Holland, Erin J Talvitie, and Michael Bowling. The effect of planning shape on dyna-style planning in high-dimensional state spaces. *arXiv preprint arXiv:1806.01825*, 2018.

Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Assaf Hallak and Shie Mannor. Consistent on-line off-policy evaluation. In *International Conference on Machine Learning*, pages 1372–1383. PMLR, 2017.

Carles Gelada and Marc G Bellemare. Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3647–3655, 2019.

Junfeng Wen, Bo Dai, Lihong Li, and Dale Schuurmans. Batch stationary distribution estimation. *arXiv preprint arXiv:2003.00722*, 2020.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019a.

Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019b.

Ziyang Tang, Yihao Feng, Lihong Li, Dengyong Zhou, and Qiang Liu. Doubly robust bias reduction in infinite horizon off-policy estimation. *arXiv preprint arXiv:1910.07186*, 2019.

Ofir Nachum and Bo Dai. Reinforcement learning via fenchel-rockafellar duality. *arXiv preprint arXiv:2001.01866*, 2020.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(4), 2010.

Ian Osband and Benjamin Van Roy. On lower bounds for regret in reinforcement learning. *arXiv preprint arXiv:1608.02732*, 2016.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2):209–232, 2002.

David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.