On-Policy Robot Imitation Learning from a Converging Supervisor

Ashwin Balakrishna*, Brijen Thananjeyan*, Jonathan Lee, Felix Li, Arsh Zahed, Joseph E. Gonzalez, Ken Goldberg
Department of Electrical Engineering and Computer Sciences
University of California Berkeley United States
ashwin_balakrishna@berkeley.edu, bthananjeyan@berkeley.edu
* equal contribution

Abstract: Existing on-policy imitation learning algorithms, such as DAgger, assume access to a fixed supervisor. However, there are many settings where the supervisor may evolve during policy learning, such as a human performing a novel task or an improving algorithmic controller. We formalize imitation learning from a "converging supervisor" and provide sublinear static and dynamic regret guarantees against the best policy in hindsight with labels from the converged supervisor, even when labels during learning are only from intermediate supervisors. We then show that this framework is closely connected to a class of reinforcement learning (RL) algorithms known as dual policy iteration (DPI), which alternate between training a reactive learner with imitation learning and a model-based supervisor with data from the learner. Experiments suggest that when this framework is applied with the state-of-the-art deep model-based RL algorithm PETS as an improving supervisor, it outperforms deep RL baselines on continuous control tasks and provides up to an 80-fold speedup in policy evaluation.

Keywords: Imitation Learning, Online Learning, Reinforcement Learning

1 Introduction

In robotics there is significant interest in using human or algorithmic supervisors to train policies via imitation learning [1, 2, 3, 4]. For example, a trained surgeon with experience teleoperating a surgical robot can provide successful demonstrations of surgical maneuvers [5]. Similarly, known dynamics models can be used by standard control techniques, such as model predictive control (MPC), to generate controls to optimize task reward [6, 7]. However, there are many cases in which the supervisor is not fixed, but is *converging* to improved behavior over time, such as when a human is initially unfamiliar with a teleoperation interface or task or when the dynamics of the system are initially unknown and estimated with experience from the environment when training an algorithmic controller. Furthermore, these supervisors are often *slow*, as humans can struggle to execute stable, high-frequency actions on a robot [7] and model-based control techniques, such as MPC, typically require computationally expensive stochastic optimization techniques to plan over complex dynamics models [8, 9, 10]. This motivates algorithms that can distill supervisors which are both *converging* and *slow* into policies that can be efficiently executed in practice. The idea of distilling improving algorithmic controllers into reactive policies has been explored in a class of reinforcement learning (RL) algorithms known as dual policy iteration (DPI) [11, 12, 13], which alternate between optimizing a reactive learner with imitation learning and a model-based supervisor with data from the learner. However, past methods have mostly been applied in discrete settings [11, 12] or make specific structural assumptions on the supervisor [13].

This paper analyzes learning from a converging supervisor in the context of on-policy imitation learning. Prior analysis of on-policy imitation learning algorithms provide regret guarantees given a fixed supervisor [14, 15, 16, 17]. We consider a converging sequence of supervisors and show that similar guarantees hold for the regret against the best policy in hindsight with labels from the converged supervisor, even when only intermediate supervisors provide labels during learning. Since

3rd Conference on Robot Learning (CoRL 2019), Osaka, Japan.

the analysis makes no structural assumptions on the supervisor, this flexibility makes it possible to use any off-policy method as the supervisor in the presented framework, such as an RL algorithm or a human, provided that it converges to a good policy on the learner's distribution. We implement an instantiation of this framework with the deep MPC algorithm PETS [8] as an improving supervisor and maintain the data efficiency of PETS while significantly reducing online computation time, accelerating both policy learning and evaluation.

The key contribution of this work is a new framework for on-policy imitation learning from a converging supervisor. We present a new notion of static and dynamic regret in this setting and provide sublinear regret guarantees by showing a reduction from this new notion of regret to the standard notion for the fixed supervisor setting. The dynamic regret result is particularly unintuitive, as it indicates that it is possible to do well on each round of learning compared to a learner with labels from the converged supervisor, even though labels are only provided by intermediate supervisors during learning. We then show that the presented framework relaxes assumptions on the supervisor in DPI and perform simulated continuous control experiments suggesting that when a PETS supervisor [8] is used, we can outperform other deep RL baselines while achieving up to an 80-fold speedup in policy evaluation. Experiments on a physical surgical robot yield up to an 20-fold reduction in query time and 53% reduction in policy evaluation time after accounting for hardware constraints.

2 Related Work

On-policy imitation learning algorithms that directly learn reactive policies from a supervisor were popularized with DAgger [18], which iteratively improves the learner by soliciting supervisor feedback on the learner's trajectory distribution. This yields significant performance gains over analogous off-policy methods [19, 20]. On-policy methods have been applied with both human [21] and algorithmic supervisors [7], but with a fixed supervisor as the guiding policy. We propose a setting where the supervisor improves over time, which is common when learning from a human or when distilling a computationally expensive, iteratively improving controller into a policy that can be efficiently executed in practice. Recently, convergence results and guarantees on regret metrics such as dynamic regret have been shown for the fixed supervisor setting [16, 17, 22]. We extend these results and present a static and dynamic analysis of on-policy imitation learning from a convergent sequence of supervisors. Recent work proposes using inverse RL to outperform an improving supervisor [23]. We instead study imitation learning in this context to use an evolving supervisor for policy learning.

Model-based planning has seen significant interest in RL due to the benefits of leveraging structure in settings such as games and robotic control [11, 12, 13]. Deep model-based reinforcement learning (MBRL) has demonstrated superior data efficiency compared to model-free methods and state-of-the-art performance on a variety of continuous control tasks [8, 9, 10]. However, these techniques are often too computationally expensive for high-frequency execution, significantly slowing down policy evaluation. To address the online burden of model-based algorithms, Sun et al. [13] define a novel class of algorithms, dual policy iteration (DPI), which alternate between optimizing a fast learner for policy evaluation using labels from a model-based supervisor and optimizing a slower model-based supervisor using trajectories from the learner. However, past work in DPI either involves planning in discrete state spaces [11, 12], or making specific assumptions on the structure of the model-based controller [13]. We discuss how the converging supervisor framework is connected to DPI, but enables a more flexible supervisor specification. We then provide a practical algorithm by using the deep MBRL algorithm PETS [8] as an improving supervisor to achieve fast policy evaluation while maintaining the data efficiency of PETS.

3 Converging Supervisor Framework and Preliminaries

3.1 On-Policy Imitation Learning

We consider continuous control problems in a finite-horizon Markov decision process (MDP), which is defined by a tuple $(\mathcal{S}, \mathcal{A}, P(\cdot,\cdot), T, R(\cdot,\cdot))$ where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. The stochastic dynamics model P maps a state s and action a to a probability distribution over states, T is the task horizon, and R is the reward function. A deterministic control policy π maps an input state in $\mathcal{S}$ to an action in $\mathcal{A}$. The goal in RL is to learn a policy π over the MDP which induces a trajectory distribution that maximizes the sum of rewards along the trajectory. In imitation

learning, this objective is simplified by instead optimizing a surrogate loss function which measures the discrepancy between the actions chosen by learned parameterized policy π_θ and supervisor ψ.

Rather than directly optimizing R from experience, on-policy imitation learning involves executing a policy in the environment and then soliciting feedback from a supervisor on the visited states. This is in contrast to off-policy methods, such as behavior cloning, in which policy learning is performed entirely on states from the supervisor's trajectory distribution. The surrogate loss of a policy π_θ along a trajectory is a supervised learning cost defined by the supervisor relabeling the trajectory's states with actions. The goal of on-policy imitation is to find the policy minimizing the corresponding surrogate risk on its own trajectory distribution. On-policy algorithms typically adhere to the following iterative procedure: (1) at iteration i, execute the current policy π_{θ_i} by deploying the learner in the MDP, observing states and actions as trajectories; (2) Receive labels for each state from the supervisor ψ; (3) Update π_{θ_i} according to the supervised learning loss to generate $\pi_{\theta_{i+1}}$.

On-policy imitation learning has often been viewed as an instance of online optimization or online learning [14, 16, 17]. Online optimization is posed as a game between an adversary, which generates a loss function l_i at iteration i and an algorithm, which plays a policy π_{θ_i} in an attempt to minimize the total incurred losses. After observing l_i, the algorithm updates its policy $\pi_{\theta_{i+1}}$ for the next iteration. In the context of imitation learning, the loss $l_i(\cdot)$ at iteration i corresponds to the supervised learning loss function under the current policy. The loss function $l_i(\cdot)$ can then be used to update the policy for the next iteration. The benefit of reducing on-policy imitation learning to online optimization is that well-studied analyses and regret metrics from online optimization can be readily applied to understand and improve imitation learning algorithms. Next, we outline a theoretical framework in which to study on-policy imitation learning with a converging supervisor.

3.2 Converging Supervisor Framework (CSF)

We begin by presenting a set of definitions for on-policy imitation learning with a converging supervisor in order to analyze the static regret (Section 4.1) and dynamic regret (Section 4.2) that can be achieved in this setting. In this paper, we assume that policies π_θ are parameterized by a parameter θ from a convex compact set $\Theta \subset \mathbb{R}^d$ equipped with the l_2-norm, which we denote with $\|\cdot\|$ for simplicity for both vectors and operators.

Definition 3.1. ***Supervisor:*** *We can think of a converging supervisor as a sequence of supervisors (labelers), $(\psi_i)_{i=1}^{\infty}$, where ψ_i defines a deterministic controller which maps an input state in $\mathcal{S}$ to an action in $\mathcal{A}$. Supervisor ψ_i provides labels for imitation learning policy updates at iteration i.*

Definition 3.2. ***Learner:*** *The learner is represented at iteration i by a parameterized policy $\pi_{\theta_i} : \mathcal{S} \to \mathcal{A}$ where π_{θ_i} is differentiable function in the policy parameter $\theta_i \in \Theta$.*

We denote the state and action at timestep t in the trajectory τ sampled at iteration i by the learner with s_t^i and a_t^i respectively.

Definition 3.3. ***Losses:*** *We consider losses at each round i of the form: $l_i(\pi_\theta, \psi_i) = \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\pi_\theta(s_t^i) - \psi_i(s_t^i)\|^2 \right]$ where $p(\tau|\theta_i)$ defines the distribution of trajectories generated by π_{θ_i}. Gradients of l_i with respect to θ are defined as $\nabla_\theta l_i(\pi_{\theta_i}, \psi_i) = \nabla_\theta l_i(\pi_\theta, \psi_i)\big|_{\theta=\theta_i}$.*

For analysis of the converging supervisor setting, we adopt the following standard assumptions. The assumptions in this section and the loss formulation are consistent with those in Hazan [24] and Ross et al. [14] for analysis of online optimization and imitation learning algorithms. The loss incurred by the agent is the population risk of the policy, and extension to empirical risk can be derived via standard concentration inequalities as in Ross et al. [14].

Assumption 3.1. ***Strongly convex losses:*** *$\forall \theta_i \in \Theta$, $l_i(\pi_\theta, \psi)$ is strongly convex with respect to θ with parameter $\alpha \in \mathbb{R}^+$. Precisely, we assume that*

$$l_i(\pi_{\theta_2}, \psi) \geq l_i(\pi_{\theta_1}, \psi) + \nabla_\theta l_i(\pi_{\theta_1}, \psi)^T(\theta_2 - \theta_1) + \frac{\alpha}{2}\|\theta_2 - \theta_1\|^2 \quad \forall\, \theta_1, \theta_2 \in \Theta$$

The expectation over $p(\tau|\theta_i)$ in Assumption 3.1 preserves strong convexity of the squared loss for an individual sample, which is assumed to be convex in θ.

Assumption 3.2. ***Bounded operator norm of policy Jacobian:*** *$\|\nabla_\theta \pi_{\theta_i}(s)\| \leq G \quad \forall s \in \mathcal{S}$, $\forall\, \theta, \theta_i \in \Theta$ where $G \in \mathbb{R}^+$.*

Assumption 3.3. ***Bounded action space:*** *The action space $\mathcal{A}$ has diameter δ. Equivalently stated:* $\delta = \sup_{a_1, a_2 \in \mathcal{A}} \|a_1 - a_2\|$.

4 Regret Analysis

We analyze the performance of well-known algorithms in on-policy imitation learning and online optimization under the converging supervisor framework. In this setting, we emphasize that the goal is to achieve low loss $l_i(\pi_{\theta_i}, \psi_N)$ with respect to labels from the last observed supervisor ψ_N. We achieve these results through regret analysis via reduction of on-policy imitation learning to online optimization, where regret is a standard notion for measuring the performance of algorithms. We consider two forms: static and dynamic regret [25], both of which have been utilized in previous on-policy imitation learning analyses [14, 16]. In this paper, regret is defined with respect to the expected losses under the trajectory distribution induced by the realized sequence of policies $(\pi_{\theta_i})_{i=1}^N$. Standard concentration inequalities can be used for finite sample analysis as in Ross et al. [14].

Using static regret, we can show a loose upper bound on average performance with respect to the last observed supervisor with minimal assumptions, similar to [14]. Using dynamic regret, we can tighten this upper bound, showing that θ_i is optimal in expectation on its own distribution with respect to ψ_N for certain algorithms, similar to [16, 22]; however, to achieve this stronger result, we require an additional continuity assumption on the dynamics of the system, which was shown to be necessary by Cheng and Boots [17]. To harness regret analysis in imitation learning, we seek to show that algorithms achieve *sublinear regret* (whether static or dynamic), denoted by $\mathcal{o}(N)$ where N is the number of iterations. That is, the regret should grow at a slower rate than linear in the number of iterations. While existing algorithms can achieve sublinear regret in the fixed supervisor setting, we analyze regret with respect to the last observed supervisor ψ_N, even though the learner is only provided labels from the intermediate ones during learning. See supplementary material for all proofs.

4.1 Static Regret

Here we show that as long as the supervisor labels are Cauchy in expectation, i.e. if $\forall s \in \mathcal{S},\ \forall N > i, \|\psi_i(s) - \psi_N(s)\| \leq f_i$ where $\lim_{i \to \infty} f_i = 0$, it is possible to achieve sublinear static regret with respect to the best policy in hindsight with labels from ψ_N for the whole dataset. This is a more difficult metric than is typically considered in regret analysis for on-policy imitation learning since labels are provided by the converging supervisor ψ_i at iteration i, but regret is evaluated with respect to the best policy given labels from ψ_N. Past work has shown that it is possible to obtain sublinear static regret in the fixed supervisor setting under strongly convex losses for standard on-policy imitation learning algorithms such as online gradient descent [24] and DAgger [14]; we extend this and show that the additional asymptotic regret in the converging supervisor setting depends only on the convergence rate of the supervisor. The standard notion of static regret is given in Definition 4.1.

Definition 4.1. *The static regret with respect to the sequence of supervisors $(\psi_i)_{i=1}^N$ is given by the difference in the performance of policy π_{θ_i} and that of the best policy in hindsight under the average trajectory distribution induced by the incurred losses with labels from current supervisor ψ_i.*

$$\text{Regret}_N^S((\psi_i)_{i=1}^N) = \sum_{i=1}^N l_i(\pi_{\theta_i}, \psi_i) - \sum_{i=1}^N l_i(\pi_{\theta^*}, \psi_i) \text{ where } \theta^* = \arg\min_{\theta \in \Theta} \sum_{i=1}^N l_i(\pi_\theta, \psi_i)$$

However, we instead analyze the more difficult regret metric presented in Definition 4.2 below.

Definition 4.2. *The static regret with respect to the supervisor ψ_N is given by the difference in the performance of policy π_{θ_i} and that of the best policy in hindsight under the average trajectory distribution induced by the incurred losses with labels from the last observed supervisor ψ_N.*

$$\text{Regret}_N^S(\psi_N) = \sum_{i=1}^N l_i(\pi_{\theta_i}, \psi_N) - \sum_{i=1}^N l_i(\pi_{\theta^\star}, \psi_N) \text{ where } \theta^\star = \arg\min_{\theta \in \Theta} \sum_{i=1}^N l_i(\pi_\theta, \psi_N)$$

Theorem 4.1. $\text{Regret}_N^S(\psi_N)$ *can be bounded above as follows:*

$$\text{Regret}_N^S(\psi_N) \leq \text{Regret}_N^S((\psi_i)_{i=1}^N) + 4\delta \sum_{i=1}^N \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^T \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right]$$

Theorem 4.1 essentially states that the expected static regret in the converging supervisor setting can be decomposed into two terms: one that is the standard notion of static regret, and an additional term that scales with the rate at which the supervisor changes. Thus, as long as there exists an algorithm to achieve sublinear static regret on the standard problem, the only additional regret comes from the evolution of the supervisor. Prior work has shown that algorithms such as online gradient descent [24] and DAgger [14] achieve sublinear static regret under strongly convex losses. Given this reduction, we see that these algorithms can also be used to achieve sublinear static regret in the converging supervisor setup if the extra term is sublinear. Corollary 4.1 identifies when this is the case.

Corollary 4.1. *If* $\forall s \in \mathcal{S},\ \forall N > i, \|\psi_i(s) - \psi_N(s)\| \leq f_i$ *where* $\lim_{i\to\infty} f_i = 0$*, then* $\text{Regret}_N^S(\psi_N)$ *can be decomposed as follows:*

$$\text{Regret}_N^S(\psi_N) = \text{Regret}_N^S((\psi_i)_{i=1}^N) + \mathcal{O}(N)$$

4.2 Dynamic Regret

Although the static regret analysis provides a bound on the average loss, the quality of that bound depends on the term $\min_\theta \sum_{i=1}^N l_i(\pi_\theta, \psi_N)$, which in practice is often very large due to approximation error between the policy class and the actual supervisor. Furthermore, it has been shown that despite sublinear static regret, policy learning may be unstable under certain dynamics [17, 21]. Recent analyses have turned to dynamic regret [16, 17], which measures the sub-optimality of a policy on its own distribution: $l_i(\pi_{\theta_i}, \psi_N) - \min_\theta l_i(\pi_\theta, \psi_N)$. Thus, low dynamic regret shows that a policy is on average performing optimally on its own distribution. This framework also helps determine if policy learning will be stable or if convergence is possible [16]. However, these notions require understanding the sensitivity of the MDP to changes in the policy. We quantify this with an additional Lipschitz assumption on the trajectory distributions induced by the policy as in [16, 17, 22]. We show that even in the converging supervisor setting, it is possible to achieve sublinear dynamic regret given this additional assumption and a converging supervisor by reducing the problem to a predictable online learning problem [22]. Note that this yields the surprising result that it is possible to do well on each round even against a dynamic comparator which has labels from the last observed supervisor. The standard notion of dynamic regret is given in Definition 4.3 below.

Definition 4.3. *The dynamic regret with respect to the sequence of supervisors* $(\psi_i)_{i=1}^N$ *is given by the difference in the performance of policy* π_{θ_i} *and that of the best policy under the current round's loss, which compares the performance of current policy* π_{θ_i} *and current supervisor* ψ_i*.*

$$\text{Regret}_N^D((\psi_i)_{i=1}^N) = \sum_{i=1}^N l_i(\pi_{\theta_i}, \psi_i) - \sum_{i=1}^N l_i(\pi_{\theta_i^*}, \psi_i) \text{ where } \theta_i^* = \arg\min_{\theta\in\Theta} l_i(\pi_\theta, \psi_i)$$

However, similar to the static regret analysis in Section 4.1, we seek to analyze the dynamic regret with respect to labels from the last observed supervisor ψ_N, which is defined as follows.

Definition 4.4. *The dynamic regret with respect to supervisor* ψ_N *is given by the difference in the performance of policy* π_{θ_i} *and that of the best policy under the current round's loss, which compares the performance of current policy* π_{θ_i} *and last observed supervisor* ψ_N*.*

$$\text{Regret}_N^D(\psi_N) = \sum_{i=1}^N l_i(\pi_{\theta_i}, \psi_N) - \sum_{i=1}^N l_i(\pi_{\theta_i^\star}, \psi_N) \text{ where } \theta_i^\star = \arg\min_{\theta\in\Theta} l_i(\pi_\theta, \psi_N)$$

We first show that there is a reduction from $\text{Regret}_N^D(\psi_N)$ to $\text{Regret}_N^D((\psi_i)_{i=1}^N)$.

Lemma 4.1. $\text{Regret}_N^D(\psi_N)$ *can be bounded above as follows:*

$$\text{Regret}_N^D(\psi_N) \leq \text{Regret}_N^D((\psi_i)_{i=1}^N) + 4\delta \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^T \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right]$$

Given the notion of supervisor convergence discussed in Corollary 4.1, Corollary 4.2 shows that if we can achieve sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$, we can also achieve sublinear $\text{Regret}_N^D(\psi_N)$.

Corollary 4.2. *If* $\forall s \in \mathcal{S},\ \forall N > i, \|\psi_i(s) - \psi_N(s)\| \leq f_i$ *where* $\lim_{i\to\infty} f_i = 0$*, then* $\text{Regret}_N^D(\psi_N)$ *can be decomposed as follows:*

$$\text{Regret}_N^D(\psi_N) = \text{Regret}_N^D((\psi_i)_{i=1}^N) + \mathcal{O}(N)$$

It is well known that $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ cannot be sublinear in general [16]. However, as in [16, 17], we can obtain conditions for sublinear regret by leveraging the structure in the imitation learning problem with a Lipschitz continuity condition on the trajectory distribution. Let $d_{TV}(p,q) = \frac{1}{2}\int |p-q| d\tau$ denote the total variation distance between two trajectory distributions p and q.

Assumption 4.1. *There exists $\eta \geq 0$ such that the following holds on the trajectory distributions induced by policies parameterized by θ_1 and θ_2:*

$$d_{TV}(p(\tau|\theta_1), p(\tau|\theta_2)) \leq \eta \|\theta_1 - \theta_2\|$$

A similar assumption is made by popular RL algorithms [26, 27], and Lemma 4.2 shows that with it, sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ can be achieved using results from predictable online learning [22].

Lemma 4.2. *If Assumption 4.1 holds and $\alpha > 4G\eta \sup_{a\in\mathcal{A}} \|a\|$, then there exists an algorithm where $\text{Regret}_N^D((\psi_i)_{i=1}^N) = o(N)$. If the diameter of the parameter space is bounded, the greedy algorithm that plays $\theta_{i+1} = \arg\min_{\theta\in\Theta} l_i(\pi_\theta, \psi_N)$ achieves sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$. Furthermore, if the losses are γ-smooth in θ and $\frac{4G\eta \sup_{a\in\mathcal{A}} \|a\|}{\alpha} > \frac{\alpha}{2\gamma}$, then online gradient descent achieves sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$.*

Finally, we combine the results of Corollary 4.2 and Lemma 4.2 to conclude that since we can achieve sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ and have found a reduction from $\text{Regret}_N^D(\psi_N)$ to $\text{Regret}_N^D((\psi_i)_{i=1}^N)$, we can also achieve sublinear dynamic regret in the converging supervisor setting.

Theorem 4.2. *If $\forall s \in \mathcal{S},\ \forall N > i, \|\psi_i(s) - \psi_N(s)\| \leq f_i$ where $\lim_{i\to\infty} f_i = 0$ and under the assumptions in Lemma 4.2, there exists an algorithm where $\text{Regret}_N^D(\psi_N) = o(N)$. If the diameter of the parameter space is bounded, the greedy algorithm that plays $\theta_{i+1} = \arg\min_{\theta\in\Theta} l_i(\pi_\theta, \psi_N)$ achieves sublinear $\text{Regret}_N^D(\psi_N)$. Furthermore, if the losses are γ-smooth in θ and $\frac{4G\eta \sup_{a\in\mathcal{A}} \|a\|}{\alpha} > \frac{\alpha}{2\gamma}$, then online gradient descent achieves sublinear $\text{Regret}_N^D(\psi_N)$.*

5 Converging Supervisors for Deep Continuous Control

Sun et al. [13] apply DPI to continuous control tasks, but assume that both the learner and supervisor are of the same policy class and from a class of distributions for which computing the KL-divergence is computationally tractable. These constraints on supervisor structure limit model capacity compared to state-of-the-art deep RL algorithms. In contrast, we do not constrain the structure of the supervisor, making it possible to use any converging, improving supervisor (algorithmic or human) with no additional engineering effort. Note that while all provided guarantees only require that the supervisor *converges*, we implicitly assume that the supervisor labels actually *improve* with respect to the MDP reward function, R, when trained with data on the learner's distribution for the learner to achieve good task performance. This assumption is validated by the experimental results in this paper and those in prior work [11, 12]. One strategy to encourage the supervisor to improve on the learner's distribution is to add noise to the learner policy to increase the variety of the experience used by the supervisor to learn information such as system dynamics. However, this was not necessary for the environments considered in this paper, and we defer further study in this direction to future work.

We utilize the converging supervisor framework (CSF) to motivate an algorithm that uses the state-of-the-art deep MBRL algorithm, PETS, as an improving supervisor. Note that while for analysis we assume a deterministic supervisor, PETS produces stochastic supervision for the agent. We observe that this does not detrimentally impact performance of the policy in practice. PETS was chosen since it has demonstrated superior data efficiency compared to other deep RL algorithms [8]. We collect policy rollouts from a model-free learner policy and refit the policy on each episode using DAgger [14] with supervision from PETS, which maintains a trained dynamics model based on the transitions collected by the learner. Supervision is generated via MPC by using the cross entropy method to plan over the learned dynamics for each state in the learner's rollout, but is collected after the rollout has completed rather than at each timestep of every policy rollout to reduce online computation time.

6 Experiments

The method presented in Section 5 uses the Converging Supervisor Framework (CSF) to train a learner policy to imitate a PETS supervisor trained on the learner's distribution. We expect the CSF learner to

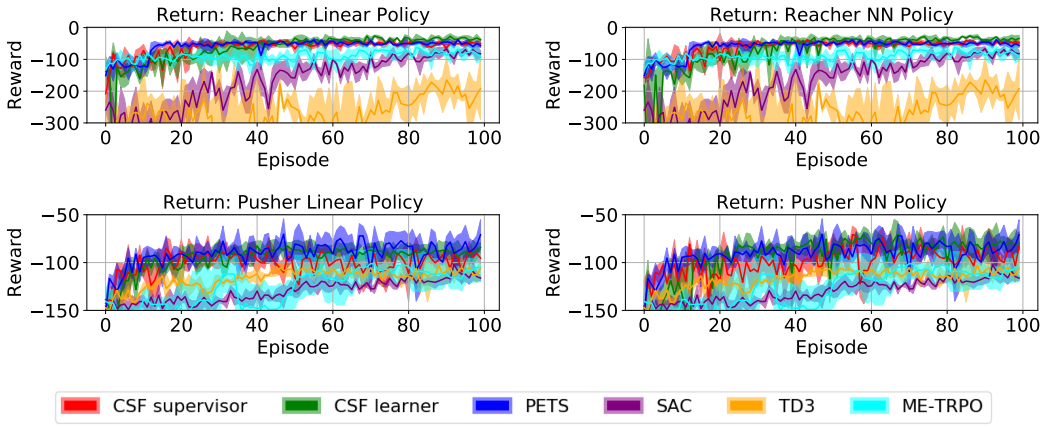

Figure 1: **Simulation experiments:** Training curves for the CSF learner, CSF supervisor, PETS, and baselines for the MuJoCo Reacher (top) and Pusher (bottom) tasks for a linear (left) and neural network (NN) policy (right). The linear policy is trained via ridge-regression with regularization parameter $\alpha = 1$, satisfying the strongly-convex loss assumption in Section 3. To test more complex policy representations, we repeat experiments with a neural network (NN) learner with 2 hidden layers with 20 hidden units each. The CSF learner successfully tracks the CSF supervisor on both domains, performs well compared to PETS, and outperforms other baselines with both policy representations. The CSF learner is slightly less data efficient than PETS, but policy evaluation is up to 80x faster than PETS. SAC, TD3, and ME-TRPO use a neural network policy/dynamics class.

be less data efficient than PETS, but have significantly faster policy evaluation time. To evaluate this hypothesis, we measure the gap in data efficiency between the learner on its own distribution (CSF learner), the supervisor on the learner's distribution (CSF supervisor) and the supervisor on its own distribution (PETS). Returns for the CSF learner and CSF supervisor are computed by rolling out the model-free learner policy and model-based controller after each training episode. Because the CSF supervisor is trained with off-policy data from the learner, the difference between the performance of the CSF learner and CSF supervisor measures how effectively the CSF learner is able to track the CSF supervisor's performance. The difference in performance between the CSF supervisor and PETS measures how important on-policy data is for PETS to generate good labels. All runs are repeated 3 times to control for stochasticity in training; see supplementary material for further experimental details. The DPI algorithm in Sun et al. [13] did not perform well on the presented environments, so we do not report a comparison to it. However, we compare against the following set of 3 state-of-the-art model-free and model-based RL baselines and demonstrate that the CSF learner maintains the data efficiency of PETS while reducing online computation time significantly by only collecting policy rollouts from the fast model-free learner instead of from the PETS supervisor.

1. **Soft Actor Critic (SAC)**: State-of-the-art maximum entropy model-free RL algorithm [28].
2. **Twin Delayed Deep Deterministic policy gradients (TD3)**: State-of-the-art model-free RL algorithm [29] which uses target networks and delayed policy updates to improve DDPG [30], a popular actor critic algorithm.
3. **Model-Ensemble Trust Region Policy Optimization (ME-TRPO)**: State-of-the-art model-free, model-based RL hybrid algorithm using a set of learned dynamics models to update a closed-loop policy offline with model-free RL [27].

6.1 Simulation Experiments

We consider the PR2 Reacher and Pusher continuous control MuJoCo domains from Chua et al. [8] (Figure 1) since these are standard benchmarks on which PETS attains good performance. For both tasks, the CSF learner outperforms other state-of-the-art deep RL algorithms, demonstrating that the CSF learner enables fast policy evaluation while maintaining data efficient learning. The CSF learner closely matches the performance of both the CSF supervisor and PETS, indicating that the CSF learner has similar data efficiency as PETS. Results using a neural network CSF learner suggest that losses strongly-convex in θ may not be necessary in practice.

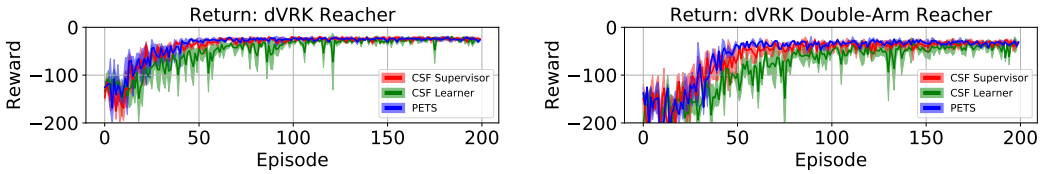

Figure 2: **Physical experiments:** Training curves for the CSF learner, CSF supervisor and PETS on the da Vinci surgical robot. The CSF learner is able to track the CSF supervisor and PETS effectively and can be queried up to 20x faster than PETS. However, due to control frequency limitations on this system, the CSF learner has a policy evaluation time that is only 1.52 and 1.46 times faster than PETS for the single and double-arm tasks respectively. The performance gap between the CSF learner and the supervisor takes longer to diminish for the harder double-arm task.

Table 1: **Policy evaluation and query times:** We report policy evaluation times in seconds over 100 episodes for the CSF learner and PETS (format: mean $\pm$ standard deviation). Furthermore, for physical experiments, we also report the total time taken to query the learner and PETS over an episode, since this difference in query times indicates the true speedup that CSF can enable (format: (total query time, policy evaluation time)). Policy evaluation and query times are nearly identical for simulation experiments. We see that the CSF learner is 20-80 times faster to query than PETS across all tasks. Results are reported on a desktop running Ubuntu 16.04 with a 3.60 GHz Intel Core i7-6850K and a NVIDIA GeForce GTX 1080. We use the NN policy for all timing results.

	PR2 Reacher (Sim)	PR2 Pusher (Sim)	dVRK Reacher	dVRK Double-Arm Reacher
CSF Learner	$\mathbf{0.29 \pm 0.01}$	$\mathbf{1.13 \pm 0.66}$	$(\mathbf{0.036 \pm 0.009}, \mathbf{5.54 \pm 0.67})$	$(\mathbf{0.038 \pm 0.0074}, \mathbf{8.87 \pm 1.12})$
PETS	24.77 ± 0.08	57.77 ± 17.12	$(0.78 \pm 0.022, 8.43 \pm 1.07)$	$(0.88 \pm 0.068, 12.97 \pm 0.77)$

This result is promising because if the model-free learner policy is able to achieve similar performance to the supervisor on its own distribution, we can simultaneously achieve the data efficiency benefits of MBRL and the low online computation time of model-free methods. To quantify this speedup, we present timing results in Table 1, which demonstrate that a significant speedup (up to 80x in this case) in policy evaluation is possible. Note that although we still need to evaluate the model-based controller on each state visited by the learner to generate labels, since this only needs to be done offline, this can be parallelized to reduce offline computation time as well.

6.2 Physical Robot Experiments

We also test CSF with a neural network policy on a physical da Vinci Surgical Robot (dVRK) [31] to evaluate its performance on multi-goal tasks where the end effector must be controlled to desired positions in the workspace. We evaluate the CSF learner/supervisor and PETS on the physical robot for both single and double arm versions of this task, and find that the CSF learner is able to track the PETS supervisor effectively (Figure 2) and provide up to a 22x speedup in policy query time (Table 1). We expect the CSF learner to demonstrate significantly greater speedups relative to standard deep MBRL for higher dimensional tasks and for systems where higher-frequency commands are possible.

7 Conclusion

We formally introduce the converging supervisor framework for on-policy imitation learning and show that under standard assumptions, we can achieve sublinear static and dynamic regret against the best policy in hindsight with labels from the last observed supervisor, even when labels are only provided by the converging supervisor during learning. We then show a connection between the converging supervisor framework and DPI, and use this to present an algorithm to accelerate policy evaluation for model-based RL without making any assumptions on the structure of the supervisor. We use the state-of-the-art deep MBRL algorithm, PETS, as an improving supervisor and maintain its data efficiency while significantly accelerating policy evaluation. Finally, we evaluate the efficiency of the method by successfully training a policy on a multi-goal reacher task directly on a physical surgical robot. The provided analysis and framework suggests a number of interesting questions regarding the degree to which non-stationary supervisors affect policy learning. In future work, it would be interesting to derive specific convergence guarantees for the converging supervisor setting, consider different notions of supervisor convergence, and study the trade-offs between supervision quality and quantity.

Acknowledgments

This research was performed at the AUTOLAB at UC Berkeley This research was performed at the AUTOLAB at UC Berkeley in affiliation with the Berkeley AI Research (BAIR) Lab, Berkeley Deep Drive (BDD), the Real-Time Intelligent Secure Execution (RISE) Lab, and the CITRIS "People and Robots" (CPAR) Initiative. The authors were supported in part by the Scalable Collaborative Human-Robot Learning (SCHooL) Project, NSF National Robotics Initiative Award 1734633 and by donations from Google, Siemens, Amazon Robotics, Toyota Research Institute, Autodesk, ABB, Samsung, Knapp, Loccioni, Honda, Intel, Comcast, Cisco, Hewlett-Packard and by equipment grants from PhotoNeo, NVidia, and Intuitive Surgical. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors. We thank our colleagues who provided helpful feedback, code, and suggestions, especially Michael Danielczuk, Anshul Ramachandran, and Ajay Tanwani.

References

[1] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. In S. Levine, V. Vanhoucke, and K. Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 357–368. PMLR, 13–15 Nov 2017. URL http://proceedings.mlr.press/v78/finn17a.html.

[2] Y. Liu, A. Gupta, P. Abbeel, and S. Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *ICRA*, pages 1118–1125. IEEE, 2018.

[3] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. In *ICLR (Workshop)*. OpenReview.net, 2018.

[4] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Y. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2018.

[5] Y. Gao, S. S. Vedula, C. E. Reiley, N. Ahmidi, B. Varadarajan, H. C. Lin, L. Tao, L. Zappella, B. Béjar, D. D. Yuh, et al. Jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling. In *MICCAI Workshop: M2CAI*, volume 3, page 3, 2014.

[6] G. Kahn, T. Zhang, S. Levine, and P. Abbeel. Plato: Policy learning using adaptive trajectory optimization. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3342–3349, 2017.

[7] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots. Agile autonomous driving via end-to-end deep imitation learning. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

[8] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *NeurIPS*, abs/1805.12114, 2018. URL http://arxiv.org/abs/1805.12114.

[9] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. *ICRA*, 2018.

[10] B. Thananjeyan, A. Balakrishna, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg. Extending deep model predictive control with safety augmented value estimation from demonstrations. *arXiv preprint arXiv:1905.13402*, 2019.

[11] T. Anthony, Z. Tian, and D. Barber. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems*, pages 5360–5370, 2017.

[12] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[13] W. Sun, G. J. Gordon, B. Boots, and J. Bagnell. Dual policy iteration. In *Advances in Neural Information Processing Systems*, pages 7059–7069, 2018.

[14] S. Ross, G. J. Gordon, and J. A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.

[15] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell. Deeply AggreVaTeD: Differentiable imitation learning for sequential prediction. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3309–3318, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[16] J. Lee, M. Laskey, A. K. Tanwani, A. Aswani, and K. Y. Goldberg. A dynamic regret analysis and adaptive regularization algorithm for on-policy robot imitation learning. *WAFR*, 2019.

[17] C. Cheng and B. Boots. Convergence of value aggregation for imitation learning. *International Conference on Artificial Intelligence and Statistics*, abs/1801.07292, 2018. URL http://arxiv.org/abs/1801.07292.

[18] S. Ross, G. J. Gordon, and J. A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *International Conference on Artificial Intelligence and Statistics*, 2011.

[19] J. A. D. Bagnell. An invitation to imitation. Technical Report CMU-RI-TR-15-08, Carnegie Mellon University, Pittsburgh, PA, March 2015.

[20] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in neural information processing systems*, pages 305–313, 1989.

[21] M. Laskey, C. Chuck, J. Lee, J. Mahler, S. Krishnan, K. Jamieson, A. Dragan, and K. Goldberg. Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 358–365. IEEE, 2017.

[22] C. Cheng, J. Lee, K. Goldberg, and B. Boots. Online learning with continuous variations: Dynamic regret and reductions. *CoRR*, abs/1902.07286, 2019.

[23] A. Jacq, M. Geist, A. Paiva, and O. Pietquin. Learning from a learner. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2990–2999, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/jacq19a.html.

[24] E. Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4): 157–325, 2016.

[25] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.

[26] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[27] T. Kurutach, I. Clavera, Y. Duan, A. Tamar, and P. Abbeel. Model-ensemble trust-region policy optimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJJinbWRZ.

[28] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm*, 2018.

[29] S. Fujimoto, H. van Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *ICML*, 2018.

[30] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2015. URL http://arxiv.org/abs/1509.02971.

[31] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio. An open-source research kit for the da Vinci surgical system. In *IEEE Intl. Conf. on Robotics and Auto. (ICRA)*, pages 6434–6439, Hong Kong, China, 2014.

[32] K. Chua. Experiment code for "deep reinforcement learning in a handful of trials using probabilistic dynamics models". https://github.com/kchua/handful-of-trials, 2018.

[33] V. Pong. rlkit. https://github.com/vitchyr/rlkit, 2018-2019.

[34] T. Kurutach. Model-ensemble trust-region policy optimization (me-trpo). https://github.com/thanard/me-trpo, 2019.

[35] D. Seita, S. Krishnan, R. Fox, S. McKinley, J. Canny, and K. Goldberg. Fast and reliable autonomous surgical debridement with cable-driven robots using a two-phase calibration procedure. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6651–6658, May 2018. doi:10.1109/ICRA.2018.8460583.

On-Policy Robot Imitation Learning from a Converging Supervisor Supplementary Material

A Static Regret

A.1 Proof of Theorem 4.1

Recall the standard notion of static regret as defined in Definition 4.1:

$$\text{Regret}_N^S((\psi_i)_{i=1}^N) = \sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_i) - l_i(\pi_{\theta^*}, \psi_i)] \text{ where } \theta^* = \arg\min_{\theta\in\Theta} \sum_{i=1}^N l_i(\pi_\theta, \psi_i) \quad (1)$$

However, we seek to bound

$$\text{Regret}_N^S(\psi_N) = \sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta^\star}, \psi_N)] \text{ where } \theta^\star = \arg\min_{\theta\in\Theta} \sum_{i=1}^N l_i(\pi_\theta, \psi_N) \quad (2)$$

as defined in Definition 4.2.

Notice that this corresponds to the static regret of the agent with respect to the losses parameterized by the last observed supervisor ψ_N. We can do this as follows:

$$\begin{aligned}
\text{Regret}_N^S(\psi_N) &= \sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta^\star}, \psi_N)] && (3)\\
&= \sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta^\star}, \psi_N)] - \text{Regret}_N^S((\psi_i)_{i=1}^N) + \text{Regret}_N^S((\psi_i)_{i=1}^N) && (4)\\
&= \sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i}, \psi_i)] + \sum_{i=1}^N [l_i(\pi_{\theta^*}, \psi_i) - l_i(\pi_{\theta^\star}, \psi_N)] \\
&\quad + \text{Regret}_N^S((\psi_i)_{i=1}^N) && (5)\\
&\leq \sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i}, \psi_i)] + \sum_{i=1}^N [l_i(\pi_{\theta^*}, \psi_i) - l_i(\pi_{\theta^\star}, \psi_N)] \\
&\quad + \text{Regret}_N^S((\psi_i)_{i=1}^N) && (6)
\end{aligned}$$

Here, inequality 6 follows from the fact that $\sum_{i=1}^N l_i(\pi_{\theta^*}, \psi_i) \leq \sum_{i=1}^N l_i(\pi_{\theta^\star}, \psi_i)$. Now, we can focus on bounding the extra term. Let $h(x, y) = \|x - y\|^2$.

$$\begin{aligned}
&\sum_{i=1}^N [l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i}, \psi_i)] + \sum_{i=1}^N [l_i(\pi_{\theta^*}, \psi_i) - l_i(\pi_{\theta^\star}, \psi_N)] && (7)\\
&= \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T h(\pi_{\theta_i}(s_t^i), \psi_N(s_t^i)) - h(\pi_{\theta_i}(s_t^i), \psi_i(s_t^i))\right] \\
&\quad + \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T h(\pi_{\theta^\star}(s_t^i), \psi_i(s_t^i)) - h(\pi_{\theta^\star}(s_t^i), \psi_N(s_t^i))\right] && (8)\\
&\leq \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T \langle \nabla_\psi h(\pi_{\theta_i}(s_t^i), \psi_N(s_t^i)), \psi_N(s_t^i) - \psi_i(s_t^i)\rangle\right] \\
&\quad + \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T \langle \nabla_\psi h(\pi_{\theta^\star}(s_t^i), \psi_i(s_t^i)), \psi_i(s_t^i) - \psi_N(s_t^i)\rangle\right] && (9)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \langle 2(\psi_N(s_t^i) - \pi_{\theta_i}(s_t)), \psi_N(s_t^i) - \psi_i(s_t^i) \rangle \right] \\
&\quad + \sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \langle 2(\psi_i(s_t^i) - \pi_{\theta^\star}(s_t)), \psi_i(s_t^i) - \psi_N(s_t^i) \rangle \right] \qquad (10) \\
&\leq \sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} 2\|\psi_N(s_t^i) - \pi_{\theta_i}(s_t)\| \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right] \\
&\quad + \sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} 2\|\psi_i(s_t^i) - \pi_{\theta^\star}(s_t)\| \|\psi_i(s_t^i) - \psi_N(s_t^i)\| \right] \qquad (11) \\
&\leq 4\delta \sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right] \qquad (12)
\end{aligned}
$$

Equation 8 follows from applying the definition of the loss function. Inequality 9 follows from applying convexity of h in ψ. Equation 10 follows from evaluating the corresponding gradients. Inequality 11 follows from Cauchy-Schwarz and inequality 12 follows from the action space bound. Thus, we have:

$$
\text{Regret}_N^S(\psi_N) \leq 4\delta \sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right] + \text{Regret}_N^S((\psi_i)_{i=1}^N) \qquad (13)
$$

A.2 Proof of Corollary 4.1

$$
\forall s \in \mathcal{S},\ \forall N > i, \|\psi_i(s) - \psi_N(s)\| \leq f_i \text{ where } \lim_{i\to\infty} f_i = 0 \qquad (14)
$$

implies that

$$
\mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_i(s_t^i) - \psi_N(s_t^i)\| \right] \leq f_i\ \forall N > i \in \mathbb{N} \qquad (15)
$$

This in turn implies that

$$
\sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_i(s_t^i) - \psi_N(s_t^i)\| \right] \leq \sum_{i=1}^{N} f_i \qquad (16)
$$

Remark: For sublinearity, we really only need inequality 15 to hold. Due to the dependence of $p(\tau|\theta_i)$ on the parameter θ_i of the policy at iteration i, we tighten this assumption with the stricter Cauchy condition 14 to remove the dependence of a component of the regret on the sequence of policies used.

The Additive Cesàro's Theorem states that if the sequence $(a_n)_{n=1}^{\infty}$ has a limit, then

$$
\lim_{n\to\infty} \frac{a_1 + a_2 \ldots a_n}{n} = \lim_{n\to\infty} a_n
$$

Thus, we see that if $\lim_{i\to\infty} f_i = 0$, then it must be the case that $\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} f_i = 0$. This shows that for some $(f_i)_{i=1}^N$ converging to 0, it must be the case that

$$
\sum_{i=1}^{N} \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_i(s_t^i) - \psi_N(s_t^i)\| \right] \leq \sum_{i=1}^{N} f_i = o(N)
$$

Thus, based on the regret bound in Theorem 4.1, we can achieve sublinear $\text{Regret}_N^S(\psi_N)$ for any sequence $(f_i)_{i=1}^N$ which converges to 0 given an algorithm that achieves sublinear $\text{Regret}_N^S((\psi_i)_{i=1}^N)$:

$$
\text{Regret}_N^S(\psi_N) = \text{Regret}_N^S((\psi_i)_{i=1}^N) + o(N)
$$

□

B Dynamic Regret

B.1 Proof of Lemma 4.1

Recall the standard notion of dynamic regret as defined in Definition 4.3:

$$\text{Regret}_N^D((\psi_i)_{i=1}^N) = \sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_i) - l_i(\pi_{\theta_i^*}, \psi_i)\right] \text{ where } \theta_i^* = \arg\min_{\theta\in\Theta} l_i(\pi_\theta, \psi_i) \tag{17}$$

However, we seek to bound

$$\text{Regret}_N^D(\psi_N) = \sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i^\star}, \psi_N)\right] \text{ where } \theta_i^\star = \arg\min_{\theta\in\Theta} l_i(\pi_\theta, \psi_N) \tag{18}$$

as defined in Definition 4.4.

Notice that this corresponds to the dynamic regret of the agent with respect to the losses parameterized by the most recent supervisor ψ_N. We can do this as follows:

$$\begin{aligned}
\text{Regret}_N^D(\psi_N) &= \sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i^\star}, \psi_N)\right] && (19)\\
&= \sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i^\star}, \psi_N)\right] - \text{Regret}_N^D((\psi_i)_{i=1}^N) \\
&\quad + \text{Regret}_N^D((\psi_i)_{i=1}^N) && (20)\\
&= \sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i}, \psi_i)\right] + \sum_{i=1}^N \left[l_i(\pi_{\theta_i^\star}, \psi_i) - l_i(\pi_{\theta_i^\star}, \psi_N)\right] \\
&\quad + \text{Regret}_N^D((\psi_i)_{i=1}^N) && (21)\\
&\leq \sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i}, \psi_i)\right] + \sum_{i=1}^N \left[l_i(\pi_{\theta_i^\star}, \psi_i) - l_i(\pi_{\theta_i^\star}, \psi_N)\right] \\
&\quad + \text{Regret}_N^D((\psi_i)_{i=1}^N) && (22)
\end{aligned}$$

Here, inequality 22 follows from the fact that $l_i(\pi_{\theta_i^\star}, \psi_i) \leq l_i(\pi_{\theta_i^\star}, \psi_i)$. Now as before, we can focus on bounding the extra term. Let $h(x, y) = \|x - y\|^2$.

$$\begin{aligned}
&\sum_{i=1}^N \left[l_i(\pi_{\theta_i}, \psi_N) - l_i(\pi_{\theta_i}, \psi_i)\right] + \sum_{i=1}^N \left[l_i(\pi_{\theta_i^\star}, \psi_i) - l_i(\pi_{\theta_i^\star}, \psi_N)\right] && (23)\\
&= \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T h(\pi_{\theta_i}(s_t^i), \psi_N(s_t^i)) - h(\pi_{\theta_i}(s_t^i), \psi_i(s_t^i))\right] \\
&\quad + \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T h(\pi_{\theta_i^\star}(s_t^i), \psi_i(s_t^i)) - h(\pi_{\theta_i^\star}(s_t^i), \psi_N(s_t^i))\right] && (24)\\
&\leq \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T \langle \nabla_\psi h(\pi_{\theta_i}(s_t^i), \psi_N(s_t^i)), \psi_N(s_t^i) - \psi_i(s_t^i)\rangle\right] \\
&\quad + \sum_{i=1}^N \mathbb{E}_{\tau\sim p(\tau|\theta_i)} \left[\frac{1}{T}\sum_{t=1}^T \langle \nabla_\psi h(\pi_{\theta_i^\star}(s_t^i), \psi_i(s_t^i)), \psi_i(s_t^i) - \psi_N(s_t^i)\rangle\right] && (25)
\end{aligned}$$

$$= \sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \langle 2(\psi_N(s_t^i) - \pi_{\theta_i}(s_t)), \psi_N(s_t^i) - \psi_i(s_t^i) \rangle \right]$$
$$+ \sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \langle 2(\psi_i(s_t^i) - \pi_{\theta_i^\star}(s_t)), \psi_i(s_t^i) - \psi_N(s_t^i) \rangle \right] \tag{26}$$

$$\leq \sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} 2\|\psi_N(s_t^i) - \pi_{\theta_i}(s_t)\| \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right]$$
$$+ \sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} 2\|\psi_i(s_t^i) - \pi_{\theta_i^\star}(s_t)\| \|\psi_i(s_t^i) - \psi_N(s_t^i)\| \right] \tag{27}$$

$$\leq 4\delta \sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right] \tag{28}$$

The steps of this proof follow as in the proof of the static regret reduction. Equation 24 follows from applying the definition of the loss function. Inequality 25 follows from applying convexity of h in ψ. Equation 26 follows from evaluating the corresponding gradients. Inequality 27 follows from Cauchy-Schwarz and inequality 28 follows from the action space bound. Combining this bound with 22, we have our desired result:

$$\text{Regret}_N^D(\psi_N) \leq 4\delta \sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_N(s_t^i) - \psi_i(s_t^i)\| \right] + \text{Regret}_N^D((\psi_i)_{i=1}^N) \tag{29}$$

□

B.2 Proof of Corollary 4.2

By Corollary 4.1,

$$\sum_{i=1}^{N} \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \left[\frac{1}{T} \sum_{t=1}^{T} \|\psi_i(s_t^i) - \psi_N(s_t^i)\| \right] = \mathcal{O}(N)$$

which implies that

$$\text{Regret}_N^D(\psi_N) = \text{Regret}_N^D((\psi_i)_{i=1}^N) + \mathcal{O}(N)$$

□

B.3 Predictability of Online Learning Problems

Next, we establish that the online learning problem defined by the losses defined in Section 3 is an (α, β)-predictable online learning problem as defined in Cheng et al. [22]. An online learning problem is (α, β)-predictable if it satisfies $\forall \theta \in \Theta$, (1) $l_i(.)$ is α strongly convex in θ, (2) $\|\nabla_\theta l_{i+1}(\pi_\theta, \psi_{i+1}) - \nabla_\theta l_i(\pi_\theta, \psi_i)\| \leq \beta \|\theta_{i+1} - \theta_i\| + \zeta_i$ where $\sum_{i=1}^N \zeta_i = \mathcal{O}(N)$. Proposition 12 in Cheng et al. [22] shows that for (α, β)-predictable problems, sublinear dynamic regret can be achieved if $\alpha > \beta$. Furthermore, Theorem 3 in Cheng et al. [22] shows that if α is sufficiently large and β sufficiently small, then sublinear dynamic regret can be achieved by online gradient descent.

Lemma B.1. *If $\forall s \in \mathcal{S}$, $\forall N > i, \|\psi_i(s) - \psi_N(s)\| \leq f_i$ where $\lim_{i \to \infty} f_i = 0$, the learning problem is $(\alpha, 4G\eta \sup_{a \in \mathcal{A}} \|a\|)$-predictable in θ: $l_i(\pi_\theta, \psi)$ is α-strongly convex by assumption and if Assumption 4.1 holds, then $l_i(\pi_\theta, \psi)$ satisfies:*

$$\|\nabla_\theta l_{i+1}(\pi_\theta, \psi_{i+1}) - \nabla_\theta l_i(\pi_\theta, \psi_i)\| \leq 4G\eta \sup_{a \in \mathcal{A}} \|a\| \|\theta_{i+1} - \theta_i\| + \zeta_i \text{ where } \sum_{i=1}^{N} \zeta_i = \mathcal{O}(N)$$

Proof of Lemma B.1 We have bounded $\text{Regret}_N^D(\psi_N)$ by the sum of $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ and a sublinear term. Now, we analyze $\text{Regret}_N^D((\psi_i)_{i=1}^N)$. We note that we can achieve sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ if the losses satisfy

$$\|\nabla_\theta l_{i+1}(\pi_\theta, \psi_{i+1}) - \nabla_\theta l_i(\pi_\theta, \psi_i)\| \leq \beta \|\theta_{i+1} - \theta_i\| + \zeta_i$$

where $\sum_{i=1}^{N} \zeta_i = \mathcal{O}(N)$ by Proposition 12 in Cheng et al. [22].

Note that for $J_\tau(\pi_\theta, \psi) = \frac{1}{T}\sum_{t=1}^{T} \|\psi(s_t) - \pi_\theta(s_t)\|^2$, we have

$$\nabla_\theta l_i(\pi_\theta, \psi) = \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \frac{1}{T} \sum_{t=1}^{T} \nabla_\theta \|\psi(s_t) - \pi_\theta(s_t)\|^2 \tag{30}$$

$$= \mathbb{E}_{\tau \sim p(\tau|\theta_i)} \nabla_\theta J_\tau(\pi_\theta, \psi) \tag{31}$$

$$= \int p(\tau|\theta_i) \nabla_\theta J_\tau(\pi_\theta, \psi) d\tau \tag{32}$$

$$\nabla_\theta J_\tau(\pi_\theta, \psi) = \frac{1}{T} \sum_{s_t \in \tau} 2 \nabla_\theta \pi_\theta(s_t)^T (\pi_\theta(s_t) - \psi(s_t)) \tag{33}$$

$$= \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T (\pi_\theta(\tau) - \psi(\tau)) \tag{34}$$

where

$$\psi(\tau) = \begin{bmatrix} \psi(s_0) \\ \vdots \\ \psi(s_T) \end{bmatrix}, \ \pi_\theta(\tau) = \begin{bmatrix} \pi_\theta(s_0) \\ \vdots \\ \pi_\theta(s_T) \end{bmatrix}, \ \nabla_\theta \pi_\theta(\tau) = \begin{bmatrix} \nabla_\theta \pi_\theta(s_0) \\ \vdots \\ \nabla_\theta \pi_\theta(s_T) \end{bmatrix} \tag{35}$$

Taking the difference of the above loss gradients, we obtain:

$$\|\nabla_\theta l_{i+1}(\pi_\theta, \psi_{i+1}) - \nabla_\theta l_i(\pi_\theta, \psi_i)\| \tag{36}$$

$$= \left\| \int p(\tau|\theta_{i+1}) \nabla_\theta J_\tau(\pi_\theta, \psi_{i+1}) d\tau - \int p(\tau|\theta_i) \nabla_\theta J_\tau(\pi_\theta, \psi_i) d\tau \right\| \tag{37}$$

$$\leq \int \|p(\tau|\theta_{i+1}) \nabla_\theta J_\tau(\pi_\theta, \psi_{i+1}) - p(\tau|\theta_i) \nabla_\theta J_\tau(\pi_\theta, \psi_i)\| d\tau \tag{38}$$

$$\begin{aligned} &= \int \Big\| \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T (p(\tau|\theta_i)\psi_i(\tau) - p(\tau|\theta_{i+1})\psi_{i+1}(\tau)) \\ &\quad + \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T (p(\tau|\theta_{i+1})\pi_\theta(\tau) - p(\tau|\theta_i)\pi_\theta(\tau)) \Big\| d\tau \end{aligned} \tag{39}$$

$$\begin{aligned} &\leq \int \left\| \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T (p(\tau|\theta_i)\psi_i(\tau) - p(\tau|\theta_{i+1})\psi_{i+1}(\tau)) \right\| d\tau \\ &\quad + \int \left\| \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T \pi_\theta(\tau) (p(\tau|\theta_{i+1}) - p(\tau|\theta_i)) \right\| d\tau \end{aligned} \tag{40}$$

$$\begin{aligned} &\leq \int \left\| \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T (p(\tau|\theta_i)\psi_i(\tau) - p(\tau|\theta_{i+1})\psi_{i+1}(\tau)) \right\| d\tau \\ &\quad + 2G \sup_{a \in \mathcal{A}} \|a\| \int |p(\tau|\theta_{i+1}) - p(\tau|\theta_i)| d\tau \end{aligned} \tag{41}$$

$$\leq \int \left\| \frac{2}{T} \nabla_\theta \pi_\theta(\tau)^T (p(\tau|\theta_i)\psi_i(\tau) - p(\tau|\theta_{i+1})\psi_{i+1}(\tau)) \right\| d\tau + 2G\eta \sup_{a \in \mathcal{A}} \|a\| \|\theta_{i+1} - \theta_i\| \tag{42}$$

$$\leq \frac{2}{T} G \int \|p(\tau|\theta_i)\psi_i(\tau) - p(\tau|\theta_{i+1})\psi_{i+1}(\tau)\| d\tau + 2G\eta \sup_{a \in \mathcal{A}} \|a\| \|\theta_{i+1} - \theta_i\| \tag{43}$$

$$\begin{aligned} &= \frac{2}{T} G \int \|p(\tau|\theta_i)\psi_i(\tau) - p(\tau|\theta_i)\psi_{i+1}(\tau) + p(\tau|\theta_i)\psi_{i+1}(\tau) - p(\tau|\theta_{i+1})\psi_{i+1}(\tau)\| d\tau \\ &\quad + 2G\eta \sup_{a \in \mathcal{A}} \|a\| \|\theta_{i+1} - \theta_i\| \end{aligned} \tag{44}$$

$$\leq \frac{2}{T}G\int \|p(\tau|\theta_i)(\psi_i(\tau)-\psi_{i+1}(\tau))\| + \|(p(\tau|\theta_i)-p(\tau|\theta_{i+1}))\psi_{i+1}(\tau)\|d\tau + 2G\eta\sup_{a\in\mathcal{A}}\|a\|\|\theta_{i+1}-\theta_i\| \tag{45}$$

$$\leq \frac{2}{T}G\int p(\tau|\theta_i)\|\psi_i(\tau)-\psi_{i+1}(\tau)\|d\tau + 4G\eta\sup_{a\in\mathcal{A}}\|a\|\|\theta_{i+1}-\theta_i\| \tag{46}$$

$$\leq 2Gf_i\int p(\tau|\theta_i)d\tau + 4G\eta\sup_{a\in\mathcal{A}}\|a\|\|\theta_{i+1}-\theta_i\| \tag{47}$$

$$\leq 2Gf_i + 4G\eta\sup_{a\in\mathcal{A}}\|a\|\|\theta_{i+1}-\theta_i\| \tag{48}$$

$$= 4G\eta\sup_{a\in\mathcal{A}}\|a\|\|\theta_{i+1}-\theta_i\| + \zeta_i \tag{49}$$

where here $\zeta_i = 2Gf_i$ and we see that $2G\sum_{i=1}^{N} f_i = o(N)$ as desired for some $(f_i)_{i=1}^N$ where $\lim_{i\to\infty} f_i = 0$ as in Corollary 4.1. Equation 37 follows from applying definitions. Equation 38 follows from the triangle inequality. Equation 39 follows from substitution of the loss gradients. Inequality 40 follows from the triangle inequality and factoring out common terms. Inequality 41 follows from subadditivity, the policy Jacobian diameter and action space bound. Inequality 42 follows from Assumption 4.1. Equation 43 follows from subadditivity of the operator norm and the policy Jacobian bound. Equation 45 follows from the triangle inequality, and equation 46 follows from the triangle inequality and Assumption 4.1. Equations 47 and 49 follow from the convergence assumption of the supervisor and the triangle inequality. □

Lemma B.2. *Assumption 3.2 implies that the loss function gradients are bounded as follows:*

$$\|\nabla_\theta l_i(\pi_\theta,\psi)\| \leq 2G\delta \quad \forall\theta,\theta_i\in\Theta,\ \forall\psi$$

Proof of Lemma B.2

$$\left\|\mathbb{E}_{\tau\sim p(\tau|\theta_i)}\left[\frac{1}{T}\sum_{t=1}^{T} 2(\nabla_\theta\pi_\theta(s_t^i))^T\left(\pi_\theta(s_t^i)-\psi_i(s_t^i)\right)\right]\right\| \leq$$

$$\mathbb{E}_{\tau\sim p(\tau|\theta_i)}\left[\frac{1}{T}\sum_{t=1}^{T}\left\|2(\nabla_\theta\pi_\theta(s_t^i))^T\left(\pi_\theta(s_t^i)-\psi_i(s_t^i)\right)\right\|\right]$$

by convexity of norms $\|\cdot\|$ and Jensen's inequality.

Then, we have that

$$\|(\nabla_\theta\pi_\theta(s))^T(\pi_\theta(s)-\psi(s))\| \leq \|\nabla_\theta\pi_\theta(s)\|\|\pi_\theta(s)-\psi(s)\| \leq G\delta \ \ \forall\theta\in\Theta,\ \forall s\in\mathcal{S},\ \forall\psi$$

due to subadditivity and the assumption that the action space diameter is bounded. Thus, we have that

$$\forall\theta,\theta_i\in\Theta,\forall\psi,\|\nabla_\theta l_i(\pi_\theta,\psi)\| \leq 2G\delta \quad \square$$

B.4 Proof of Lemma 4.2

From Lemma B.1, the loss gradients are bounded by the sum of a Lipschitz-type term and a sublinear term, satisfying the conditions for Proposition 12 from Cheng et al. [22]. Thus, by Proposition 12 from Cheng et al. [22], we see that as long as $\alpha > 4G\eta\sup_{a\in\mathcal{A}}\|a\|$, there exists an algorithm that can achieve sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$. An example of an algorithm that achieves sublinear dynamic regret under this condition is the greedy algorithm [22]: $\theta_{i+1} = \arg\min_{\theta\in\Theta} l_i(\pi_\theta,\psi_i)$.

Define $\beta = 4G\eta\sup_{a\in\mathcal{A}}\|a\|$, $\lambda = \beta/\alpha$, and $\xi_i = \zeta_i/\alpha$. For the greedy algorithm, the result can be shown in a similar fashion to Theorem 3 of Cheng et al. [22]:

$$\|\theta_i^* - \theta_i\| = \|\theta_i^* - \theta_{i-1}^*\| \leq \lambda\|\theta_i-\theta_{i-1}\| + \frac{\zeta_i}{\alpha} \leq \lambda^i\|\theta_1-\theta_0\| + \sum_{j=1}^{i}\lambda^{i-j}\xi_j$$

where the first inequality follows from Proposition 1 of Lee et al. [16] and the second inequality follows from repeated application of the same proposition. Summing from 1 to N with $\zeta_i = 2Gf_i$ as

in the proof of Lemma 4.2, we have

$$\sum_{i=1}^{N}\sum_{j=1}^{i}\lambda^{i-j}\xi_j \leq \sum_{i=1}^{N}\xi_i(1+\lambda+\lambda^2+\ldots) \leq \frac{1}{1-\lambda}\sum_{i=1}^{N}\xi_i = \frac{2G}{\alpha(1-\lambda)}\sum_{i=1}^{N}f_i$$

Thus, if $\sum_{i=1}^{N} f_i = o(N)$, we can show that the greedy algorithm achieves sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ by using the Lipschitz continuity of the losses as shown in the proof of Lemma B.2 if the parameter space diameter is bounded as follows: $D = \sup_{\theta,\theta' \in \Theta} \|\theta - \theta'\|$.

$$\begin{aligned}
\text{Regret}_N^D((\psi_i)_{i=1}^N) &\leq 2G\delta \sum_{i=1}^{N} \|\theta_i - \theta_i^*\| \\
&\leq 2G\delta \left(D\sum_{i=1}^{N}\lambda^i + \frac{2G}{\alpha(1-\lambda)}\sum_{i=1}^{N} f_i \right) \\
&\leq 2G\delta \left(\frac{D}{1-\lambda} + \frac{2G}{\alpha(1-\lambda)}\sum_{i=1}^{N} f_i \right) \\
&= o(N)
\end{aligned}$$

For the last part of the lemma, the fact that online gradient descent achieves sublinear $\text{Regret}_N^D((\psi_i)_{i=1}^N)$ follows directly from applying Theorem 3 from Cheng et al. [22] with $\frac{4G\eta \sup_{a\in\mathcal{A}}\|a\|}{\alpha} > \frac{\alpha}{2\gamma}$ if the losses are γ-smooth in θ. □

B.5 Proof of Theorem 4.2

The proof follows immediately from combining the result of Corollary 4.2 and Lemma 4.2. □

C Training Details

C.1 CSF Learner

For the linear policy, the CSF learner is trained via linear regression with regularization parameter $\alpha = 1$. For the neural network policy, the CSF learner is represented with an ensemble of 5 neural networks, each with 1 layer with 20 hidden units and swish activations.

C.2 PETS

PETS learns an ensemble of neural network dynamics models using sampled transitions and updates them on-policy to better reflect the dynamics local to the learned policy's state distribution. We use the implementation from [32]. MPC is run over the learned dynamics to select actions for the next iteration. For all environments, a probabilistic ensemble of 5 neural networks with 3 hidden layers, each with 200 hidden units and swish activations are used to represent the dynamics model. The TS-∞ sampling procedure is used for planning. We use an MPC planning horizon of length 25 for all environments and 1 initial random rollout to seed the learned dynamics model. Chua et al. [8] contains further details on training PETS.

C.3 SAC

We use the rlkit implementation [33] of soft actor critic with the following parameters: batch size = 128, discount factor = 0.99, soft target τ = 0.001, policy learning rate = 0.0003, Q function learning rate = 0.0003, value function learning rate = 0.0003, and replay buffer size = 1000000. All networks are two-layer multi-layer perceptrons with 300 hidden units.

C.4 TD3

We use the rlkit implementation [33] of TD3 with the following parameters: batch size = 128, discount factor = 0.99, and replay buffer size = 1000000. The exploration strategy consists of adding Gaussian

noise $\mathcal{N}(0,\ 0.1)$ to actions chosen by the policy. All networks are two-layer multi-layer perceptrons with 300 hidden units.

C.5 ME-TRPO

We model both the policy and dynamics with neural networks, using an ensemble of dynamics models to avoid exploitation of model bias. We use the ME-TRPO implementation from [34] with the following hyperparameters: batch size=128, discount factor=1, and learning rate =.001 for both the policy and dynamics. The policy network has two hidden layers with 64 units each and all dynamics networks have two hidden layers with 512 units each and ReLU activation.

D Experimental Details

D.1 Simulated Experiments

Both simulated experiments involve manipulation tasks on a simulated PR2 robot and are from the provided code in Chua et al. [8]. Both are implemented as 7-DOF torque control tasks. For all tasks, we plot the sum of rewards for each training episode.

D.2 Physical Experiments

Both physical experiments involve delta-position control in 3D space on the daVinci surgical system, which is cable driven and hard to precisely control, making it difficult to reliably reach a desired pose without appropriate compensation [35]. The CSF learner policy and supervisor dynamics are modeled by 3 hidden-layer feed-forward neural networks with 200 hidden units each. The tasks involve guiding the end effectors to targets in the workspace and isotropic concave quadratic rewards are used. For all tasks, we plot the sum of rewards for each training episode. For multi-arm experiments, the arms are limited to subsets of the state space where collisions are not possible. We are investigating modeling arm collisions for future work. Since the da Vinci surgical system has relatively limited control frequency, although the CSF learner often enables significantly faster query time than PETS, the improvement in policy evaluation time was somewhat less significant due to physical hardware constraints. In future work, we plan to implement the proposed algorithm on a robot with higher frequency control capability.