Safe Policy Search for Lifelong Reinforcement Learning with Sublinear Regret

Haitham Bou Ammar HAITHAMB@SEAS.UPENN.EDU
Rasul Tutunov TUTUNOV@SEAS.UPENN.EDU
Eric Eaton EEATON@CIS.UPENN.EDU
University of Pennsylvania, Computer and Information Science Department, Philadelphia, PA 19104 USA

Abstract

Lifelong reinforcement learning provides a promising framework for developing versatile agents that can accumulate knowledge over a lifetime of experience and rapidly learn new tasks by building upon prior knowledge. However, current lifelong learning methods exhibit non-vanishing regret as the amount of experience increases, and include limitations that can lead to suboptimal or unsafe control policies. To address these issues, we develop a lifelong policy gradient learner that operates in an adversarial setting to learn multiple tasks online while enforcing safety constraints on the learned policies. We demonstrate, for the first time, *sublinear regret* for lifelong policy search, and validate our algorithm on several benchmark dynamical systems and an application to quadrotor control.

1. Introduction

Reinforcement learning (RL) (Busoniu et al., 2010; Sutton & Barto, 1998) often requires substantial experience before achieving acceptable performance on individual control problems. One major contributor to this issue is the *tabula-rasa* assumption of typical RL methods, which learn from scratch on each new task. In these settings, learning performance is directly correlated with the quality of the acquired samples. Unfortunately, the amount of experience necessary for high-quality performance increases exponentially with the tasks' degrees of freedom, inhibiting the application of RL to high-dimensional control problems.

When data is in limited supply, transfer learning can significantly improve model performance on new tasks by reusing previous learned knowledge during training (Taylor & Stone, 2009; Gheshlaghi Azar et al., 2013; Lazaric, 2011; Ferrante et al., 2008; Bou Ammar et al., 2012). Multi-task learning (MTL) explores another notion of knowledge transfer, in which task models are trained simultaneously and share knowledge during the joint learning process (Wilson et al., 2007; Zhang et al., 2008).

In the *lifelong learning* setting (Thrun & O'Sullivan, 1996a;b), which can be framed as an online MTL problem, agents acquire knowledge incrementally by learning multiple tasks consecutively over their lifetime. Recently, based on the work of Ruvolo & Eaton (2013) on supervised lifelong learning, Bou Ammar et al. (2014) developed a lifelong learner for policy gradient RL. To ensure efficient learning over consecutive tasks, these works employ a second-order Taylor expansion around the parameters that are (locally) optimal for each task without transfer. This assumption simplifies the MTL objective into a weighted quadratic form for online learning, but since it is based on single-task learning, this technique can lead to parameters far from globally optimal. Consequently, the success of these methods for RL highly depends on the policy initializations, which must lead to near-optimal trajectories for meaningful updates. Also, since their objective functions average loss over all tasks, these methods exhibit non-vanishing regrets of the form $\mathcal{O}(R)$, where R is the total number of rounds in a non-adversarial setting.

In addition, these methods may produce control policies with unsafe behavior (i.e., capable of causing damage to the agent or environment, catastrophic failure, etc.). This is a critical issue in robotic control, where unsafe control policies can lead to physical damage or user injury. This problem is caused by using constraint-free optimization over the shared knowledge during the transfer process, which may lead to uninformative or unbounded policies.

In this paper, we address these issues by proposing the first *safe lifelong learner* for policy gradient RL operating in an adversarial framework. Our approach rapidly learns high-performance *safe control policies* based on the agent's previously learned knowledge and safety constraints on each task, accumulating knowledge over multiple consecutive tasks to optimize overall performance. We theoretically analyze the regret exhibited by our algorithm, showing *sublinear* dependency of the form $\mathcal{O}(\sqrt{R})$ for R rounds, thus outperforming current methods. We then evaluate our approach empirically on a set of dynamical systems.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

2. Background

2.1. Reinforcement Learning

An RL agent sequentially chooses actions to minimize its expected cost. Such problems are formalized as Markov decision processes (MDPs) $\langle \mathcal{X}, \mathcal{U}, \mathcal{P}, \boldsymbol{c}, \gamma \rangle$, where $\mathcal{X} \subset \mathbb{R}^d$ is the (potentially infinite) state space, $\mathcal{U} \in \mathbb{R}^{d_a}$ is the set of all possible actions, $\mathcal{P} : \mathcal{X} \times \mathcal{U} \times \mathcal{X} \to [0,1]$ is a state transition probability describing the system's dynamics, $\boldsymbol{c} : \mathcal{X} \times \mathcal{U} \times \mathcal{X} \to \mathbb{R}$ is the cost function measuring the agent's performance, and $\gamma \in [0,1]$ is a discount factor. At each time step m, the agent is in state $\boldsymbol{x}_m \in \mathcal{X}$ and must choose an action $\boldsymbol{u}_m \in \mathcal{U}$, transitioning it to a new state $\boldsymbol{x}_{m+1} \sim \mathcal{P}(\boldsymbol{x}_{m+1}|\boldsymbol{x}_m, \boldsymbol{u}_m)$ and yielding a cost $\boldsymbol{c}_{m+1} = \boldsymbol{c}(\boldsymbol{x}_{m+1}, \boldsymbol{u}_m, \boldsymbol{x}_m)$. The sequence of state-action pairs forms a trajectory $\boldsymbol{\tau} = [\boldsymbol{x}_{0:M-1}, \boldsymbol{u}_{0:M-1}]$ over a (possibly infinite) horizon M. A policy $\pi : \mathcal{X} \times \mathcal{U} \to [0,1]$ specifies a probability distribution over state-action pairs, where $\pi(\boldsymbol{u}|\boldsymbol{x})$ represents the probability of selecting an action $\boldsymbol{u}$ in state $\boldsymbol{x}$. The goal of RL is to find an optimal policy $\pi^\star$ that minimizes the total expected cost.

Policy search methods have shown success in solving high-dimensional problems, such as robotic control (Kober & Peters, 2011; Peters & Schaal, 2008a; Sutton et al., 2000). These methods represent the policy $\pi_{\boldsymbol{\alpha}}(\boldsymbol{u}|\boldsymbol{x})$ using a vector $\boldsymbol{\alpha} \in \mathbb{R}^d$ of control parameters. The optimal policy $\pi^\star$ is found by determining the parameters $\boldsymbol{\alpha}^\star$ that minimize the expected average cost:

$$l(\boldsymbol{\alpha}) = \sum_{k=1}^{n} p_{\boldsymbol{\alpha}}\left(\boldsymbol{\tau}^{(k)}\right) \boldsymbol{C}\left(\boldsymbol{\tau}^{(k)}\right) \quad , \tag{1}$$

where n is the total number of trajectories, and $p_{\boldsymbol{\alpha}}\left(\boldsymbol{\tau}^{(k)}\right)$ and $\boldsymbol{C}\left(\boldsymbol{\tau}^{(k)}\right)$ are the probability and cost of trajectory $\boldsymbol{\tau}^{(k)}$:

$$p_{\boldsymbol{\alpha}}\left(\boldsymbol{\tau}^{(k)}\right) = \mathcal{P}_0\left(\boldsymbol{x}_0^{(k)}\right) \prod_{m=0}^{M-1} \mathcal{P}\left(\boldsymbol{x}_{m+1}^{(k)}|\boldsymbol{x}_m^{(k)}, \boldsymbol{u}_m^{(k)}\right) \times \pi_{\boldsymbol{\alpha}}\left(\boldsymbol{u}_m^{(k)}|\boldsymbol{x}_m^{(k)}\right) \tag{2}$$

$$\boldsymbol{C}\left(\boldsymbol{\tau}^{(k)}\right) = \frac{1}{M} \sum_{m=0}^{M-1} \boldsymbol{c}\left(\boldsymbol{x}_{m+1}^{(k)}, \boldsymbol{u}_m^{(k)}, \boldsymbol{x}_m^{(k)}\right) \quad , \tag{3}$$

with an initial state distribution $\mathcal{P}_0 : \mathcal{X} \to [0,1]$. We handle a constrained version of policy search, in which optimality not only corresponds to minimizing the total expected cost, but also to ensuring that the policy satisfies safety constraints. These constraints vary between applications, for example corresponding to maximum joint torque or prohibited physical positions.

2.2. Online Learning & Regret Analysis

In this paper, we employ a special form of *regret minimization games*, which we briefly review here. A regret minimization game is a triple $\langle \mathcal{K}, \mathcal{F}, R \rangle$, where $\mathcal{K}$ is a nonempty decision set, $\mathcal{F}$ is the set of moves of the adversary which contains bounded convex functions from $\mathbb{R}^n$ to $\mathbb{R}$, and R is the total number of rounds. The game proceeds in rounds, where at each round $j = 1, \ldots, R$, the agent chooses a prediction $\boldsymbol{\theta}_j \in \mathcal{K}$ and the environment (i.e., the adversary) chooses a loss function $l_j \in \mathcal{F}$. At the end of the round, the loss function l_j is revealed to the agent and the decision $\boldsymbol{\theta}_j$ is revealed to the environment. In this paper, we handle the full-information case, where the agent may observe the entire loss function l_j as its feedback and can exploit this in making decisions. The goal is to minimize the cumulative regret $\sum_{j=1}^{R} l_j(\boldsymbol{\theta}_j) - \inf_{\boldsymbol{u} \in \mathcal{K}} \left[\sum_{j=1}^{R} l_j(\boldsymbol{u})\right]$. When analyzing the regret of our methods, we use a variant of this definition to handle the lifelong RL case:

$$\mathfrak{R}_R = \sum_{j=1}^{R} l_{t_j}(\boldsymbol{\theta}_j) - \inf_{u \in \mathcal{K}} \left[\sum_{j=1}^{R} l_{t_j}(\boldsymbol{u})\right] \quad ,$$

where $l_{t_j}(\cdot)$ denotes the loss of task t at round j.

For our framework, we adopt a variant of regret minimization called "Follow the Regularized Leader," which minimizes regret in two steps. First, the unconstrained solution $\tilde{\boldsymbol{\theta}}$ is determined (see Sect. 4.1) by solving an unconstrained optimization over the accumulated losses observed so far. Given $\tilde{\boldsymbol{\theta}}$, the constrained solution is then determined by learning a projection into the constraint set via Bregman projections (see Abbasi-Yadkori et al. (2013)).

3. Safe Lifelong Policy Search

We adopt a lifelong learning framework in which the agent learns multiple RL tasks consecutively, providing it the opportunity to transfer knowledge between tasks to improve learning. Let $\mathcal{T}$ denote the set of tasks, each element of which is an MDP. At any time, the learner may face any previously seen task, and so must strive to maximize its performance across all tasks. The goal is to learn optimal policies $\pi^\star_{\boldsymbol{\alpha}_1^\star}, \ldots, \pi^\star_{\boldsymbol{\alpha}_{|\mathcal{T}|}^\star}$ for all tasks, where policy $\pi^\star_{\boldsymbol{\alpha}_t^\star}$ for task t is parameterized by $\boldsymbol{\alpha}_t^\star \in \mathbb{R}^d$. In addition, each task is equipped with safety constraints to ensure acceptable policy behavior: $\boldsymbol{A}_t \boldsymbol{\alpha}_t \leq \boldsymbol{b}_t$, with $\boldsymbol{A}_t \in \mathbb{R}^{d \times d}$ and $\boldsymbol{b}_t \in \mathbb{R}^d$ representing the allowed policy combinations. The precise form of these constraints depends on the application domain, but this formulation supports constraints on (e.g.) joint torque, acceleration, position, etc.

At each round j, the learner observes a set of n_{t_j} trajectories $\left\{\boldsymbol{\tau}_{t_j}^{(1)}, \ldots, \boldsymbol{\tau}_{t_j}^{(n_{t_j})}\right\}$ from a task $t_j \in \mathcal{T}$, where each trajectory has length M_{t_j}. To support knowledge transfer between tasks, we assume that each task's policy parameters $\boldsymbol{\alpha}_{t_j} \in \mathbb{R}^d$ at round j can be written as a linear combination of a shared latent basis $\boldsymbol{L} \in \mathbb{R}^{d \times k}$ with coefficient vectors $\boldsymbol{s}_{t_j} \in \mathbb{R}^k$; therefore, $\boldsymbol{\alpha}_{t_j} = \boldsymbol{L}\boldsymbol{s}_{t_j}$. Each column of $\boldsymbol{L}$ represents a chunk of transferrable knowledge; this task construction has been used successfully in previous

multi-task learning work (Kumar & Daumé III, 2012; Ruvolo & Eaton, 2013; Bou Ammar et al., 2014). Extending this previous work, we ensure that the shared knowledge repository is "informative" by incorporating bounding constraints on the Frobenius norm $\|\cdot\|_{\mathsf{F}}$ of $\boldsymbol{L}$. Consequently, the optimization problem after observing r rounds is:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \sum_{j=1}^{r} \left[\eta_{t_j} l_{t_j}\left(\boldsymbol{L}\boldsymbol{s}_{t_j}\right)\right] + \mu_1 \|\boldsymbol{S}\|_{\mathsf{F}}^2 + \mu_2 \|\boldsymbol{L}\|_{\mathsf{F}}^2 \qquad (4)$$
$$\text{s.t.} \quad \boldsymbol{A}_{t_j}\boldsymbol{\alpha}_{t_j} \leq \boldsymbol{b}_{t_j} \;\; \forall t_j \in \mathcal{I}_r$$
$$\boldsymbol{\lambda}_{\min}\left(\boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}\right) \geq p \;\text{ and }\; \boldsymbol{\lambda}_{\max}\left(\boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}\right) \leq q \;\; ,$$

where p and q are the constraints on $\|\boldsymbol{L}\|_{\mathsf{F}}$, $\eta_{t_j} \in \mathbb{R}$ are design weighting parameters[1], $\mathcal{I}_r = \{t_1, \ldots, t_r\}$ denotes the set of all tasks observed so far through round r, and $\boldsymbol{S}$ is the collection of all coefficients

$$\boldsymbol{S}(:,h) = \begin{cases} \boldsymbol{s}_{t_h} & \text{if } t_h \in \mathcal{I}_r \\ 0 & \text{otherwise} \end{cases} \quad \forall h \in \{1, \ldots, |\mathcal{T}|\} \;\; .$$

The loss function $l_{t_j}(\boldsymbol{\alpha}_{t_j})$ in Eq. (4) corresponds to a policy gradient learner for task t_j, as defined in Eq. (1). Typical policy gradient methods (Kober & Peters, 2011; Sutton et al., 2000) maximize a lower bound of the expected cost $l_{t_j}\left(\boldsymbol{\alpha}_{t_j}\right)$, which can be derived by taking the logarithm and applying Jensen's inequality:

$$\log\left[l_{t_j}\left(\boldsymbol{\alpha}_{t_j}\right)\right] = \log\left[\sum_{k=1}^{n_{t_j}} p_{\boldsymbol{\alpha}_{t_j}}^{(t_j)}\left(\boldsymbol{\tau}_{t_j}^{(k)}\right) \boldsymbol{C}^{(t_j)}\left(\boldsymbol{\tau}_{t_j}^{(k)}\right)\right] \qquad (5)$$
$$\geq \log\left[n_{t_j}\right] + \mathbb{E}\left[\sum_{m=0}^{M_{t_j}-1} \log\left[\pi_{\boldsymbol{\alpha}_{t_j}}\left(\boldsymbol{u}_m^{(k,t_j)} \mid \boldsymbol{x}_m^{(k,t_j)}\right)\right]\right]_{k=1}^{n_{t_j}} + \text{const} \;\; .$$

Therefore, our goal is to minimize the following objective:

$$\boldsymbol{e}_r = \sum_{j=1}^{r}\left(-\frac{\eta_{t_j}}{n_{t_j}} \sum_{k=1}^{n_{t_j}} \sum_{m=0}^{M_{t_j}-1} \log\left[\pi_{\boldsymbol{\alpha}_{t_j}}\left(\boldsymbol{u}_m^{(k,t_j)} \mid \boldsymbol{x}_m^{(k,t_j)}\right)\right]\right) \qquad (6)$$
$$+ \mu_1 \|\boldsymbol{S}\|_{\mathsf{F}}^2 + \mu_2 \|\boldsymbol{L}\|_{\mathsf{F}}^2$$
$$\text{s.t.} \quad \boldsymbol{A}_{t_j}\boldsymbol{\alpha}_{t_j} \leq \boldsymbol{b}_{t_j} \;\; \forall t_j \in \mathcal{I}_r$$
$$\boldsymbol{\lambda}_{\min}\left(\boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}\right) \geq p \;\text{ and }\; \boldsymbol{\lambda}_{\max}\left(\boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}\right) \leq q \;\; .$$

3.1. Online Formulation

The optimization problem above can be mapped to the standard online learning framework by unrolling $\boldsymbol{L}$ and $\boldsymbol{S}$ into a vector $\boldsymbol{\theta} = [\text{vec}(\boldsymbol{L})\ \text{vec}(\boldsymbol{S})]^{\mathsf{T}} \in \mathbb{R}^{dk+k|\mathcal{T}|}$. Choosing $\boldsymbol{\Omega}_0(\boldsymbol{\theta}) = \mu_2 \sum_{i=1}^{dk} \boldsymbol{\theta}_i^2 + \mu_1 \sum_{i=dk+1}^{dk+k|\mathcal{T}|} \boldsymbol{\theta}_i^2$, and $\boldsymbol{\Omega}_j(\boldsymbol{\theta}) = \boldsymbol{\Omega}_{j-1}(\boldsymbol{\theta}) + \eta_{t_j} l_{t_j}(\boldsymbol{\theta})$, we can write the safe lifelong policy search problem (Eq. (6)) as:

$$\boldsymbol{\theta}_{r+1} = \arg\min_{\boldsymbol{\theta} \in \mathcal{K}} \boldsymbol{\Omega}_r(\boldsymbol{\theta}) \;\; , \qquad (7)$$

where $\mathcal{K} \subseteq \mathbb{R}^{dk+k|\mathcal{T}|}$ is the set of allowable policies under the given safety constraints. Note that the loss for task t_j can be written as a bilinear product in $\boldsymbol{\theta}$:

$$l_{t_j}(\boldsymbol{\theta}) = -\frac{1}{n_{t_j}} \sum_{k=1}^{n_{t_j}} \sum_{m=0}^{M_{t_j}-1} \log\left[\pi_{\boldsymbol{\Theta}_{\boldsymbol{L}}\boldsymbol{\Theta}_{\boldsymbol{s}_{t_j}}}^{(t_j)}\left(\boldsymbol{u}_m^{(k,\,t_j)} \mid \boldsymbol{x}_m^{(k,\,t_j)}\right)\right]$$

$$\boldsymbol{\Theta}_{\boldsymbol{L}} = \begin{bmatrix} \boldsymbol{\theta}_1 & \ldots & \boldsymbol{\theta}_{d(k-1)+1} \\ \vdots & \vdots & \vdots \\ \boldsymbol{\theta}_d & \ldots & \boldsymbol{\theta}_{dk} \end{bmatrix}, \; \boldsymbol{\Theta}_{\boldsymbol{s}_{t_j}} = \begin{bmatrix} \boldsymbol{\theta}_{dk+1} \\ \vdots \\ \boldsymbol{\theta}_{(d+1)k+1} \end{bmatrix} .$$

We see that the problem in Eq. (7) is equivalent to Eq. (6) by noting that at r rounds, $\boldsymbol{\Omega}_r = \sum_{j=1}^{r} \eta_{t_j} l_{t_j}(\boldsymbol{\theta}) + \boldsymbol{\Omega}_0(\boldsymbol{\theta})$.

4. Online Learning Method

We solve Eq. (7) in two steps. First, we determine the unconstrained solution $\tilde{\boldsymbol{\theta}}_{r+1}$ when $\mathcal{K} = \mathbb{R}^{dk+k|\mathcal{T}|}$ (see Sect. 4.1). Given $\tilde{\boldsymbol{\theta}}_{r+1}$, we derive the constrained solution $\hat{\boldsymbol{\theta}}_{r+1}$ by learning a projection $\text{Proj}_{\boldsymbol{\Omega}_r,\mathcal{K}}\left(\tilde{\boldsymbol{\theta}}_{r+1}\right)$ to the constraint set $\mathcal{K} \subseteq \mathbb{R}^{dk+k|\mathcal{T}|}$, which amounts to minimizing the Bregman divergence over $\boldsymbol{\Omega}_r(\boldsymbol{\theta})$ (see Sect. 4.2)[2]. The complete approach is given in Algorithm 1 and is available as a software implementation on the authors' websites.

4.1. Unconstrained Policy Solution

Although Eq. (6) is not jointly convex in both $\boldsymbol{L}$ and $\boldsymbol{S}$, it is separably convex (for log-concave policy distributions). Consequently, we follow an alternating optimization approach, first computing $\boldsymbol{L}$ while holding $\boldsymbol{S}$ fixed, and then updating $\boldsymbol{S}$ given the acquired $\boldsymbol{L}$. We detail this process for two popular PG learners, eREINFORCE (Williams, 1992) and eNAC (Peters & Schaal, 2008b). The derivations of the update rules below can be found in Appendix A.

These updates are governed by learning rates β and λ that decay over time; β and λ can be chosen using line-search methods as discussed by Boyd & Vandenberghe (2004). In our experiments, we adopt a simple yet effective strategy, where $\beta = cj^{-1}$ and $\lambda = cj^{-1}$, with $0 < c < 1$.

Step 1: Updating $\boldsymbol{L}$ Holding $\boldsymbol{S}$ fixed, the latent repository can be updated according to:

$$\boldsymbol{L}_{\beta+1} = \boldsymbol{L}_{\beta} - \eta_{\boldsymbol{L}}^{\beta} \nabla_{\boldsymbol{L}} \boldsymbol{e}_r(\boldsymbol{L}, \boldsymbol{S}) \qquad \text{(eREINFORCE)}$$
$$\boldsymbol{L}_{\beta+1} = \boldsymbol{L}_{\beta} - \eta_{\boldsymbol{L}}^{\beta} \boldsymbol{G}^{-1}(\boldsymbol{L}_{\beta}, \boldsymbol{S}_{\beta}) \nabla_{\boldsymbol{L}} \boldsymbol{e}_r(\boldsymbol{L}, \boldsymbol{S}) \qquad \text{(eNAC)}$$

with learning rate $\eta_{\boldsymbol{L}}^{\beta} \in \mathbb{R}$, and $\boldsymbol{G}^{-1}(\boldsymbol{L}, \boldsymbol{S})$ as the inverse of the Fisher information matrix (Peters & Schaal, 2008b).

In the special case of Gaussian policies, the update for $\boldsymbol{L}$

[1] We describe later how to set the η's later in Sect. 5 to obtain regret bounds, and leave them as variables now for generality.

[2] In Sect. 4.2, we linearize the loss around the constrained solution of the previous round to increase stability and ensure convergence. Given the linear losses, it suffices to solve the Bregman divergence over the regularizer, reducing the computational cost.

can be derived in a closed form as $\boldsymbol{L}_{\beta+1} = \boldsymbol{Z}_{\boldsymbol{L}}^{-1}\boldsymbol{v}_{\boldsymbol{L}}$, where

$$\boldsymbol{Z}_{\boldsymbol{L}} = 2\mu_2 \boldsymbol{I}_{dk\times dk} + \sum_{j=1}^{r} \frac{\eta_{t_j}}{n_{t_j}\sigma_{t_j}^2} \sum_{k=1}^{n_{t_j}} \sum_{m=0}^{M_{t_j}-1} \text{vec}\left(\boldsymbol{\Phi}\boldsymbol{s}_{t_j}^{\mathsf{T}}\right)\left(\boldsymbol{\Phi}^{\mathsf{T}} \otimes \boldsymbol{s}_{t_j}^{\mathsf{T}}\right)$$

$$\boldsymbol{v}_{\boldsymbol{L}} = \sum_{j} \frac{\eta_{t_j}}{n_{t_j}\sigma_{t_j}^2} \sum_{k=1}^{n_{t_j}} \sum_{m=0}^{M_{t_j}-1} \text{vec}\left(\boldsymbol{u}_m^{(k,\,t_j)} \boldsymbol{\Phi}\boldsymbol{s}_{t_j}^{\mathsf{T}}\right) \ ,$$

$\sigma_{t_j}^2$ is the covariance of the Gaussian policy for a task t_j, and $\boldsymbol{\Phi} = \boldsymbol{\Phi}\left(\boldsymbol{x}_m^{(k,\,t_j)}\right)$ denotes the state features.

Step 2: Updating $\boldsymbol{S}$ Given the fixed basis $\boldsymbol{L}$, the coefficient matrix $\boldsymbol{S}$ is updated column-wise for all $t_j \in \mathcal{I}_r$:

$$\boldsymbol{s}_{\lambda+1}^{(t_j)} = \boldsymbol{s}_{\lambda+1}^{(t_j)} - \eta_{\boldsymbol{S}}^{\lambda} \nabla_{\boldsymbol{s}_{t_j}} e_r(\boldsymbol{L}, \boldsymbol{S}) \qquad \text{(eREINFORCE)}$$

$$\boldsymbol{s}_{\lambda+1}^{(t_j)} = \boldsymbol{s}_{\lambda+1}^{(t_j)} - \eta_{\boldsymbol{S}}^{\lambda} \boldsymbol{G}^{-1}(\boldsymbol{L}_\beta, \boldsymbol{S}_\beta) \nabla_{\boldsymbol{s}_{t_j}} e_r(\boldsymbol{L}, \boldsymbol{S}) \qquad \text{(eNAC)}$$

with learning rate $\eta_{\boldsymbol{S}}^{\lambda} \in \mathbb{R}$. For Gaussian policies, the closed-form of the update is $\boldsymbol{s}_{t_j} = \boldsymbol{Z}_{\boldsymbol{s}_{t_j}}^{-1}\boldsymbol{v}_{\boldsymbol{s}_{t_j}}$, where

$$\boldsymbol{Z}_{\boldsymbol{s}_{t_j}} = 2\mu_1 \boldsymbol{I}_{k\times k} + \sum_{t_k=t_j} \frac{\eta_{t_j}}{n_{t_j}\sigma_{t_j}^2} \sum_{k=1}^{n_{t_j}} \sum_{m=0}^{M_{t_j}-1} \boldsymbol{L}^{\mathsf{T}}\boldsymbol{\Phi}\boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{L}$$

$$\boldsymbol{v}_{t_j} = \sum_{t_k=t_j} \frac{\eta_{t_j}}{n_{t_j}\sigma_{t_j}^2} \sum_{k=1}^{n_{t_j}} \sum_{m=0}^{M_{t_j}-1} \boldsymbol{u}_m^{(k,\,t_j)} \boldsymbol{L}^{\mathsf{T}}\boldsymbol{\Phi} \ .$$

4.2. Constrained Policy Solution

Once we have obtained the unconstrained solution $\tilde{\boldsymbol{\theta}}_{r+1}$ (which satisfies Eq. (7), but can lead to policy parameters in unsafe regions), we then derive the constrained solution to ensure safe policies. We learn a projection $\text{Proj}_{\boldsymbol{\Omega}_r,\mathcal{K}}\left(\tilde{\boldsymbol{\theta}}_{r+1}\right)$ from $\tilde{\boldsymbol{\theta}}_{r+1}$ to the constraint set:

$$\hat{\boldsymbol{\theta}}_{r+1} = \arg\min_{\boldsymbol{\theta}\in\mathcal{K}} \mathcal{B}_{\boldsymbol{\Omega}_r,\mathcal{K}}\left(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}_{r+1}\right) \ , \tag{8}$$

where $\mathcal{B}_{\boldsymbol{\Omega}_r,\mathcal{K}}\left(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}_{r+1}\right)$ is the Bregman divergence over $\boldsymbol{\Omega}_r$:

$$\mathcal{B}_{\boldsymbol{\Omega}_r,\mathcal{K}}\left(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}_{r+1}\right) = \boldsymbol{\Omega}_r(\boldsymbol{\theta}) - \boldsymbol{\Omega}_r(\tilde{\boldsymbol{\theta}}_{r+1}) - \text{trace}\left(\nabla_{\boldsymbol{\theta}}\boldsymbol{\Omega}_r\left(\boldsymbol{\theta}\right)\Big|_{\tilde{\boldsymbol{\theta}}_{r+1}}\left(\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}}_{r+1}\right)\right) \ .$$

Solving Eq. (8) is computationally expensive since $\boldsymbol{\Omega}_r(\boldsymbol{\theta})$ includes the sum back to the original round. To remedy this problem, ensure the stability of our approach, and guarantee that the constrained solutions for all observed tasks lie within a bounded region, we linearize the current-round loss function $l_{t_r}(\boldsymbol{\theta})$ around the *constrained* solution of the previous round $\hat{\boldsymbol{\theta}}_r$:

$$l_{t_r}\left(\hat{\boldsymbol{u}}\right) = \hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}^{\mathsf{T}} \hat{\boldsymbol{u}} \ , \tag{9}$$

where

$$\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r} = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} l_{t_r}\left(\boldsymbol{\theta}\right)\Big|_{\hat{\boldsymbol{\theta}}_r} \\ l_{t_r}\left(\boldsymbol{\theta}\right)\Big|_{\hat{\boldsymbol{\theta}}_r} - \nabla_{\boldsymbol{\theta}} l_{t_r}\left(\boldsymbol{\theta}\right)\Big|_{\hat{\boldsymbol{\theta}}_r} \hat{\boldsymbol{\theta}}_r \end{bmatrix}, \quad \hat{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u} \\ 1 \end{bmatrix} .$$

Given the above linear form, we can rewrite the optimization problem in Eq. (8) as:

$$\hat{\boldsymbol{\theta}}_{r+1} = \arg\min_{\boldsymbol{\theta}\in\mathcal{K}} \mathcal{B}_{\boldsymbol{\Omega}_0,\mathcal{K}}\left(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}_{r+1}\right) \ . \tag{10}$$

Consequently, determining *safe policies* for lifelong policy search reinforcement learning amounts to solving:

$$\begin{aligned} \min_{\boldsymbol{L},\boldsymbol{S}} \ & \mu_1\|\boldsymbol{S}\|_{\mathsf{F}}^2 + \boldsymbol{\mu}_2\|\boldsymbol{L}\|_{\mathsf{F}}^2 \\ & + 2\mu_1 \text{trace}\left(\boldsymbol{S}^{\mathsf{T}}\Big|_{\tilde{\theta}_{r+1}}\boldsymbol{S}\right) + 2\mu_2 \text{trace}\left(\boldsymbol{L}\Big|_{\tilde{\theta}_{r+1}}\boldsymbol{L}\right) \\ \text{s.t. } & \boldsymbol{A}_{t_j}\boldsymbol{L}\boldsymbol{s}_{t_j} \le \boldsymbol{b}_{t_j} \quad \forall t_j \in \mathcal{I}_r \\ & \boldsymbol{L}\boldsymbol{L}^{\mathsf{T}} \le p\boldsymbol{I} \quad \text{and} \quad \boldsymbol{L}\boldsymbol{L}^{\mathsf{T}} \ge q\boldsymbol{I} \ . \end{aligned}$$

To solve the optimization problem above, we start by converting the inequality constraints to equality constraints by introducing slack variables $\boldsymbol{c}_{t_j} \ge 0$. We also guarantee that these slack variables are bounded by incorporating $\|\boldsymbol{c}_{t_j}\| \le \boldsymbol{c}_{\max}$, $\forall t_j \in \{1, \ldots, |\mathcal{T}|\}$:

$$\begin{aligned} \min_{\boldsymbol{L},\boldsymbol{S},\boldsymbol{C}} \ & \mu_1\|\boldsymbol{S}\|_{\mathsf{F}}^2 + \mu_2\|\boldsymbol{L}\|_{\mathsf{F}}^2 \\ & + 2\mu_2 \text{trace}\left(\boldsymbol{L}^{\mathsf{T}}\Big|_{\tilde{\theta}_{r+1}}\boldsymbol{L}\right) + 2\mu_1 \text{trace}\left(\boldsymbol{S}^{\mathsf{T}}\Big|_{\tilde{\theta}_{r+1}}\boldsymbol{S}\right) \\ \text{s.t. } & \boldsymbol{A}_{t_j}\boldsymbol{L}\boldsymbol{s}_{t_j} = \boldsymbol{b}_{t_j} - \boldsymbol{c}_{t_j} \quad \forall t_j \in \mathcal{I}_r \\ & \boldsymbol{c}_{t_j} > 0 \quad \text{and} \quad \|\boldsymbol{c}_{t_j}\|_2 \le \boldsymbol{c}_{\max} \quad \forall t_j \in \mathcal{I}_r \\ & \boldsymbol{L}\boldsymbol{L}^{\mathsf{T}} \le p\boldsymbol{I} \quad \text{and} \quad \boldsymbol{L}\boldsymbol{L}^{\mathsf{T}} \ge q\boldsymbol{I} \ . \end{aligned}$$

With this formulation, learning $\text{Proj}_{\boldsymbol{\Omega}_r,\mathcal{K}}\left(\tilde{\boldsymbol{\theta}}_{r+1}\right)$ amounts to solving second-order cone and semi-definite programs.

4.2.1. Semi-Definite Program for Learning $\boldsymbol{L}$

This section determines the constrained projection of the shared basis $\boldsymbol{L}$ given fixed $\boldsymbol{S}$ and $\boldsymbol{C}$. We show that $\boldsymbol{L}$ can be acquired efficiently, since this step can be relaxed to solving a semi-definite program in $\boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}$ (Boyd & Vandenberghe, 2004). To formulate the semi-definite program, note that

$$\begin{aligned} \text{trace}\left(\boldsymbol{L}^{\mathsf{T}}\Big|_{\tilde{\theta}_{r+1}}\boldsymbol{L}\right) &= \sum_{i=1}^{k} \boldsymbol{l}_{r+1}^{(i)\,\mathsf{T}}\Big|_{\tilde{\theta}_{r+1}}\boldsymbol{l}_i \\ &\le \sum_{i=1}^{k} \left\|\boldsymbol{l}_{r+1}^{(i)}\Big|_{\tilde{\theta}_{r+1}}\right\|_2 \|\boldsymbol{l}_i\|_2 \\ &\le \sqrt{\sum_{i=1}^{k}\left\|\boldsymbol{l}_r^{(i)}\Big|_{\tilde{\theta}_{r+1}}\right\|_2^2}\sqrt{\sum_{i=1}^{k}||\boldsymbol{l}_i||_2^2} \\ &= \left\|\boldsymbol{L}\Big|_{\tilde{\theta}_{r+1}}\right\|_{\mathsf{F}} \sqrt{\text{trace}\left(\boldsymbol{L}\boldsymbol{L}^{\mathsf{T}}\right)} \ . \end{aligned}$$

From the constraint set, we recognize:

$$\begin{aligned} & \boldsymbol{s}_{t_j}^{\mathsf{T}}\boldsymbol{L}^{\mathsf{T}} = \left(b_{t_j} - \boldsymbol{c}_{t_j}\right)^{\mathsf{T}}\left(\boldsymbol{A}_{t_j}^{\dagger}\right)^{\mathsf{T}} \\ \implies & \boldsymbol{s}_{t_j}^{\mathsf{T}}\boldsymbol{L}^{\mathsf{T}}\boldsymbol{L}\boldsymbol{s}_{t_j} = \boldsymbol{a}_{t_j}^{\mathsf{T}}\boldsymbol{a}_{t_j} \quad \text{with} \quad \boldsymbol{a}_{t_j} = \boldsymbol{A}_{t_j}^{\dagger}\left(\boldsymbol{b}_{t_j} - \boldsymbol{c}_{t_j}\right) \ . \end{aligned}$$

Algorithm 1 Safe Online Lifelong Policy Search

Inputs: Total number of rounds R, weighting factor $\eta = 1/\sqrt{R}$, regularization parameters μ_1 and μ_2, constraints p and q, number of latent basis vectors k.
$\boldsymbol{S} = \text{zeros}(k, |\mathcal{T}|)$, $\boldsymbol{L} = \text{diag}_k(\zeta)$ with $p \leq \zeta^2 \leq q$
for $j = 1$ to R **do**
  $t_j \leftarrow$ sampleTask(), and update $\mathcal{I}_j$
  Compute **unconstrained solution** $\tilde{\boldsymbol{\theta}}_{j+1}$ (Sect. 4.1)
  Fix $\boldsymbol{S}$ and $\boldsymbol{C}$, and update $\boldsymbol{L}$ (Sect. 4.2.1)
  Use updated $\boldsymbol{L}$ to derive $\boldsymbol{S}$ and $\boldsymbol{C}$ (Sect. 4.2.2)
end for
Output: Safety-constrained $\boldsymbol{L}$ and $\boldsymbol{S}$

Since spectrum $\left(\boldsymbol{L}\boldsymbol{L}^\mathsf{T}\right) = \text{spectrum}\left(\boldsymbol{L}^\mathsf{T}\boldsymbol{L}\right)$, we can write:

$$\min_{\boldsymbol{X}\subset\mathcal{S}_{++}} \mu_2 \text{trace}(\boldsymbol{X}) + 2\mu_2 \left|\left|\boldsymbol{L}\Big|_{\tilde{\boldsymbol{\theta}}_{r+1}}\right|\right|_\mathsf{F} \sqrt{\text{trace}\left(\boldsymbol{X}\right)}$$
$$\text{s.t. } \boldsymbol{s}_{t_j}^\mathsf{T}\boldsymbol{X}\boldsymbol{s}_{t_j} = \boldsymbol{a}_{t_j}^\mathsf{T}\boldsymbol{a}_{t_j} \quad \forall t_j \in \mathcal{I}_r$$
$$\boldsymbol{X} \leq p\boldsymbol{I} \quad \text{and} \quad \boldsymbol{X} \geq q\boldsymbol{I}\,, \quad \text{with} \quad \boldsymbol{X} = \boldsymbol{L}^\mathsf{T}\boldsymbol{L}\,.$$

4.2.2. Second-Order Cone Program for Learning Task Projections

Having determined $\boldsymbol{L}$, we can acquire $\boldsymbol{S}$ and update $\boldsymbol{C}$ by solving a second-order cone program (Boyd & Vandenberghe, 2004) of the following form:

$$\min_{\boldsymbol{s}_{t_1},\ldots,\boldsymbol{s}_{t_j},\boldsymbol{c}_{t_1},\ldots,\boldsymbol{c}_{t_j}} \mu_1 \sum_{j=1}^{r} \|\boldsymbol{s}_{t_j}\|_2^2 + 2\mu_1 \sum_{j=1}^{r} \boldsymbol{s}_{t_j}^\mathsf{T}\Big|_{\hat{\boldsymbol{\theta}}_r} \boldsymbol{s}_{t_j}$$
$$\text{s.t.} \quad \boldsymbol{A}_{t_j}\boldsymbol{L}\boldsymbol{s}_{t_j} = \boldsymbol{b}_{t_j} - \boldsymbol{c}_{t_j}$$
$$\boldsymbol{c}_{t_j} > 0 \quad \|\boldsymbol{c}_{t_j}\|_2^2 \leq \boldsymbol{c}_{\max}^2 \;\forall t_j \in \mathcal{I}_r \;\;.$$

5. Theoretical Guarantees

This section quantifies the performance of our approach by providing formal analysis of the regret after R rounds. We show that the safe lifelong reinforcement learner exhibits *sublinear* regret in the total number of rounds. Formally, we prove the following theorem:

Theorem 1 (Sublinear Regret)**.** *After R rounds and choosing $\forall t_j \in \mathcal{I}_R \;\; \eta_{t_j} = \eta = \frac{1}{\sqrt{R}}$, $\boldsymbol{L}\Big|_{\hat{\boldsymbol{\theta}}_1} = diag_{\boldsymbol{k}}(\zeta)$, with $diag_{\boldsymbol{k}}(\cdot)$ being a diagonal matrix among the $\boldsymbol{k}$ columns of $\boldsymbol{L}$, $p \leq \zeta^2 \leq q$, and $\boldsymbol{S}\Big|_{\hat{\boldsymbol{\theta}}_1} = \boldsymbol{0}_{k\times|\mathcal{T}|}$, the safe lifelong reinforcement learner exhibits sublinear regret of the form:*

$$\sum_{j=1}^{R} l_{t_j}\left(\hat{\boldsymbol{\theta}}_j\right) - l_{t_j}(\boldsymbol{u}) = \mathcal{O}\left(\sqrt{R}\right) \;\; \textit{for any } \boldsymbol{u} \in \mathcal{K}.$$

Proof Roadmap: The remainder of this section completes our proof of Theorem 1; further details are given in Appendix B. We assume linear losses for all tasks in the constrained case in accordance with Sect. 4.2. Although linear losses for policy search RL are too restrictive given a single operating point, as discussed previously, we remedy this problem by generalizing to the case of piece-wise linear losses, where the linearization operating point is a resultant of the optimization problem. To bound the regret, we need to bound the dual Euclidean norm (which is the same as the Euclidean norm) of the gradient of the loss function, then prove Theorem 1 by bounding: (1) task t_j's gradient loss (Sect. 5.1), and (2) linearized losses with respect to $\boldsymbol{L}$ and $\boldsymbol{S}$ (Sect. 5.2).

5.1. Bounding t_j's Gradient Loss

We start by stating essential lemmas for Theorem 1; due to space constraints, proofs for all lemmas are available in the supplementary material. Here, we bound the gradient of a loss function $l_{t_j}(\boldsymbol{\theta})$ at round r under Gaussian policies[3].

Assumption 1. *We assume that the policy for a task t_j is Gaussian, the action set $\mathcal{U}$ is bounded by $\boldsymbol{u}_{\max}$, and the feature set is upper-bounded by $\boldsymbol{\Phi}_{\max}$.*

Lemma 1. *Assume task t_j's policy at round r is given by $\pi_{\boldsymbol{\alpha}_{t_j}}^{(t_j)}\left(\boldsymbol{u}_m^{(k,\,t_j)}|\boldsymbol{x}_m^{(k,\,t_j)}\right)\Big|_{\hat{\boldsymbol{\theta}}_r} = \mathcal{N}\left(\boldsymbol{\alpha}_{t_j}^\mathsf{T}\Big|_{\hat{\boldsymbol{\theta}}_r}\boldsymbol{\Phi}\left(\boldsymbol{x}_m^{(k,\,t_j)}\right), \boldsymbol{\sigma}_{t_j}\right)$, for states $\boldsymbol{x}_m^{(k,\,t_j)} \in \mathcal{X}_{t_j}$ and actions $\boldsymbol{u}_m^{(k,\,t_j)} \in \mathcal{U}_{t_j}$. For $l_{t_j}(\boldsymbol{\alpha}_{t_j}) = -\frac{1}{n_{t_j}}\sum_{k=1}^{n_{t_j}}\sum_{m=0}^{M_{t_j}-1}\log\left[\pi_{\boldsymbol{\alpha}_{t_j}}^{(t_j)}\left(\boldsymbol{u}_m^{(k,\,t_j)}|\boldsymbol{x}_m^{(k,\,t_j)}\right)\right]$, the gradient $\nabla_{\boldsymbol{\alpha}_{t_j}} l_{t_j}(\boldsymbol{\alpha}_{t_j})\Big|_{\hat{\boldsymbol{\theta}}_r}$ satisfies $\left|\left|\nabla_{\boldsymbol{\alpha}_{t_j}} l_{t_j}(\boldsymbol{\alpha}_{t_j})\Big|_{\hat{\boldsymbol{\theta}}_r}\right|\right|_2 \leq \frac{M_{t_j}}{\sigma_{t_j}^2}\left(u_{\max} + \max_{t_k\in\mathcal{I}_{r-1}}\left\{\left|\left|\boldsymbol{A}_{t_k}^+\right|\right|_2\left(||\boldsymbol{b}_{t_k}||_2 + \boldsymbol{c}_{\max}\right)\right\}\boldsymbol{\Phi}_{\max}\right)\boldsymbol{\Phi}_{\max}$ for all trajectories and all tasks, with $u_{\max} = \max_{k,m}\left\{\left|\boldsymbol{u}_m^{(k,\,t_j)}\right|\right\}$ and $\boldsymbol{\Phi}_{\max} = \max_{k,m}\left\{\left|\left|\boldsymbol{\Phi}\left(\boldsymbol{x}_m^{(k,\,t_j)}\right)\right|\right|_2\right\}$.*

5.2. Bounding Linearized Losses

As discussed previously, we linearize the loss of task t_r around the constraint solution of the previous round $\hat{\boldsymbol{\theta}}_r$. To acquire the regret bounds in Theorem 1, the next step is to bound the dual norm, $\left|\left|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right|\right|_2^\star = \left|\left|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right|\right|_2$ of Eq. (9). It can be easily seen

$$\left|\left|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right|\right|_2 \leq \underbrace{\left|l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right|}_{\text{constant}} + \underbrace{\left|\left|\nabla_{\boldsymbol{\theta}} l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right|\right|_2}_{\text{Lemma 2}} + \left|\left|\nabla_{\boldsymbol{\theta}} l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right|\right|_2 \times \underbrace{\left|\left|\hat{\boldsymbol{\theta}}_r\right|\right|_2}_{\text{Lemma 3}}\,. \tag{11}$$

[3]Please note that derivations for other forms of log-concave policy distributions could be derived in similar manner. In this work, we focus on Gaussian policies since they cover a broad spectrum of real-world applications.

Since $\left|l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right|$ can be bounded by $\boldsymbol{\delta}_{l_{t_r}}$ (see Sect. 2), the next step is to bound $\left\|\nabla_{\boldsymbol{\theta}} l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2$, and $\|\hat{\boldsymbol{\theta}}_r\|_2$.

Lemma 2. *The norm of the gradient of the loss function evaluated at $\hat{\boldsymbol{\theta}}_r$ satisfies*

$$\left\|\nabla_{\boldsymbol{\theta}} l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2^2 \leq \left\|\nabla_{\boldsymbol{\alpha}_{t_r}} l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2^2 \Bigg(q \times d \left({}^{2d}/_{p^2} \max_{t_k \in \mathcal{I}_{r-1}} \left\{ \left\|\boldsymbol{A}_{t_k}^{\dagger}\right\|_2^2 \left(||\boldsymbol{b}_{t_k}||_2^2 + \boldsymbol{c}_{max}^2 \right) \right\} + 1 \right) \Bigg) \ .$$

To finalize the bound of $\left\|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2$ as needed for deriving the regret, we must derive an upper-bound for $\|\hat{\boldsymbol{\theta}}_r\|_2$:

Lemma 3. *The L_2 norm of the constraint solution at round $r-1$, $\|\hat{\boldsymbol{\theta}}_r\|_2^2$ is bounded by*

$$\|\hat{\boldsymbol{\theta}}_r\|_2^2 \leq q \times d \Bigg[1 + |\mathcal{I}_{r-1}| \frac{1}{p^2} \max_{t_k \in \mathcal{I}_{r-1}} \left\{ \left\|\boldsymbol{A}_{t_k}^{\dagger}\right\|_2^2 \left(||\boldsymbol{b}_{t_k}||_2 + \boldsymbol{c}_{max} \right)^2 \right\} \Bigg] \ ,$$

where $|\mathcal{I}_{r-1}|$ is the number of unique tasks observed so far.

Given the previous two lemmas, we can prove the bound for $\left\|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2$:

Lemma 4. *The L_2 norm of the linearizing term of $l_{t_r}(\boldsymbol{\theta})$ around $\hat{\boldsymbol{\theta}}_r$, $\left\|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2$, is bounded by*

$$\begin{aligned} \left\|\hat{\boldsymbol{f}}_{t_r}\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2 &\leq \left\|\nabla_{\boldsymbol{\theta}} l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2 \left(1 + \|\hat{\boldsymbol{\theta}}_r\|_2\right) + \left|l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right| \quad (12) \\ &\leq \boldsymbol{\gamma}_1(r)\left(1 + \boldsymbol{\gamma}_2(r)\right) + \boldsymbol{\delta}_{l_{t_r}} \ , \end{aligned}$$

where $\boldsymbol{\delta}_{l_{t_r}}$ is the constant upper-bound on $\left|l_{t_r}(\boldsymbol{\theta})\Big|_{\hat{\boldsymbol{\theta}}_r}\right|$, and

$$\boldsymbol{\gamma}_1(r) = \frac{1}{n_{t_j}\sigma_{t_j}^2} \Bigg[\Bigg(u_{max} + \max_{t_k \in \mathcal{I}_{r-1}} \left\{ \left|\left|\boldsymbol{A}_{t_k}^{+}\right|\right|_2 \left(||\boldsymbol{b}_{t_k}||_2 + \boldsymbol{c}_{max} \right) \right\} \boldsymbol{\Phi}_{max} \Bigg) \boldsymbol{\Phi}_{max} \Bigg] \times \left(\frac{d}{p}\sqrt{2q} \sqrt{\max_{t_k \in \mathcal{I}_{r-1}} \left\{ \|\boldsymbol{A}_{t_k}^{\dagger}\|_2^2 \left(\|\boldsymbol{b}_{t_k}\|_2^2 + \boldsymbol{c}_{max}^2 \right) \right\}} + \sqrt{qd} \right)$$

$$\boldsymbol{\gamma}_2(r) \leq \sqrt{q \times d} + \sqrt{|\mathcal{I}_{r-1}|} \sqrt{1 + \frac{1}{p^2} \max_{t_k \in \mathcal{I}_{r-1}} \left\{ \left\|\boldsymbol{A}_{t_k}^{\dagger}\right\|_2^2 \left(||\boldsymbol{b}_{t_k}||_2 + \boldsymbol{c}_{max} \right)^2 \right\}} \ .$$

5.3. Completing the Proof of Sublinear Regret

Given the lemmas in the previous section, we now can derive the sublinear regret bound given in Theorem 1. Using results developed by Abbasi-Yadkori et al. (2013), it is easy to see that

$$\nabla_{\boldsymbol{\theta}} \boldsymbol{\Omega}_0\left(\tilde{\boldsymbol{\theta}}_j\right) - \nabla_{\boldsymbol{\theta}} \boldsymbol{\Omega}_0\left(\tilde{\boldsymbol{\theta}}_{j+1}\right) = \eta_{t_j} \hat{\boldsymbol{f}}_{t_j}\Big|_{\hat{\boldsymbol{\theta}}_j} \ .$$

From the convexity of the regularizer, we obtain:

$$\boldsymbol{\Omega}_0\left(\hat{\boldsymbol{\theta}}_j\right) \geq \boldsymbol{\Omega}_0\left(\hat{\boldsymbol{\theta}}_{j+1}\right) + \left\langle \nabla_{\boldsymbol{\theta}} \boldsymbol{\Omega}_0\left(\hat{\boldsymbol{\theta}}_{j+1}\right), \hat{\boldsymbol{\theta}}_j - \hat{\boldsymbol{\theta}}_{j+1} \right\rangle + \frac{1}{2}\left\|\hat{\boldsymbol{\theta}}_j - \hat{\boldsymbol{\theta}}_{j+1}\right\|_2^2 \ .$$

We have:

$$\left\|\hat{\boldsymbol{\theta}}_j - \hat{\boldsymbol{\theta}}_{j+1}\right\|_2 \leq \eta_{t_j} \left\|\hat{\boldsymbol{f}}_{t_j}\Big|_{\hat{\boldsymbol{\theta}}_j}\right\|_2 \ .$$

Therefore, for any $\boldsymbol{u} \in \mathcal{K}$

$$\sum_{j=1}^{r} \eta_{t_j}\left(l_{t_j}\left(\hat{\boldsymbol{\theta}}_j\right) - l_{t_j}(\boldsymbol{u})\right) \leq \sum_{j=1}^{r} \eta_{t_j} \left\|\hat{\boldsymbol{f}}_{t_j}\Big|_{\hat{\boldsymbol{\theta}}_j}\right\|_2^2 + \boldsymbol{\Omega}_0(\boldsymbol{u}) - \boldsymbol{\Omega}_0(\hat{\boldsymbol{\theta}}_1) \ .$$

Assuming that $\forall t_j \ \eta_{t_j} = \eta$, we can derive:

$$\sum_{j=1}^{r}\left(l_{t_j}\left(\hat{\boldsymbol{\theta}}_j\right) - l_{t_j}(\boldsymbol{u})\right) \leq \eta \sum_{j=1}^{r} \left\|\hat{\boldsymbol{f}}_{t_j}\Big|_{\hat{\boldsymbol{\theta}}_j}\right\|_2^2 + {}^{1}/_{\eta}\left(\boldsymbol{\Omega}_0(\boldsymbol{u}) - \boldsymbol{\Omega}_0(\hat{\boldsymbol{\theta}}_1)\right) \ .$$

The following lemma finalizes the proof of Theorem 1:

Lemma 5. *After R rounds with $\forall t_j \ \eta_{t_j} = \eta = \frac{1}{\sqrt{R}}$, for any $\boldsymbol{u} \in \mathcal{K}$ we have that $\sum_{j=1}^{R} l_{t_j}(\hat{\boldsymbol{\theta}}_j) - l_{t_j}(\boldsymbol{u}) \leq \mathcal{O}\left(\sqrt{R}\right)$.*

Proof. From Eq. (12), it follows that

$$\begin{aligned} \left\|\hat{\boldsymbol{f}}_{t_j}\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2^2 &\leq \boldsymbol{\gamma}_3(R) + 4\boldsymbol{\gamma}_1^2(R)\boldsymbol{\gamma}_2^2(R) \\ &\leq \boldsymbol{\gamma}_3(R) + 8\frac{d}{p^2}\boldsymbol{\gamma}_1^2(R) qd \Bigg(1 + |\mathcal{I}_{R-1}| \times \max_{t_k \in \mathcal{I}_{R-1}} \left\{ \|\boldsymbol{A}_{t_k}^{\dagger}\|_2 \left(\|\boldsymbol{b}_{t_k}\|_2 + \boldsymbol{c}_{\max} \right)^2 \right\} \Bigg) \end{aligned}$$

with $\boldsymbol{\gamma}_3(R) = 4\boldsymbol{\gamma}_1^2(R) + 2\max_{t_j \in \mathcal{I}_{R-1}} \boldsymbol{\delta}_{t_j}^2$. Since $|\mathcal{I}_{R-1}| \leq |\mathcal{T}|$, we have that $\left\|\hat{\boldsymbol{f}}_{t_j}\Big|_{\hat{\boldsymbol{\theta}}_r}\right\|_2^2 \leq \boldsymbol{\gamma}_5(R)|\mathcal{T}|$ with $\boldsymbol{\gamma}_5 = 8^{d}/_{p^2} q \boldsymbol{\gamma}_1^2(R) \max_{t_k \in \mathcal{I}_{R-1}} \left\{ \|\boldsymbol{A}_{t_k}^{\dagger}\|_2^2 \left(\|\boldsymbol{b}_{t_k}\|_2 + \boldsymbol{c}_{\max} \right)^2 \right\}$.

Given that $\boldsymbol{\Omega}_0(\boldsymbol{u}) \leq qd + \boldsymbol{\gamma}_5(R)|\mathcal{T}|$, with $\boldsymbol{\gamma}_5(R)$ being a constant, we have:

$$\sum_{j=1}^{r}\left(l_{t_j}\left(\hat{\boldsymbol{\theta}}_j\right) - l_{t_j}(\boldsymbol{u})\right) \leq \eta \sum_{j=1}^{r} \boldsymbol{\gamma}_5(R)|\mathcal{T}| + \frac{1}{\eta}\left(qd + \boldsymbol{\gamma}_5(R)|\mathcal{T}| - \boldsymbol{\Omega}_0(\hat{\boldsymbol{\theta}}_1)\right) \ .$$

Initializing $\boldsymbol{L}$ and $\boldsymbol{S}$: We initialize $\boldsymbol{L}\Big|_{\hat{\boldsymbol{\theta}}_1} = \text{diag}_{\boldsymbol{k}}(\zeta)$, with $p \leq \zeta^2 \leq q$ and $\boldsymbol{S}\Big|_{\hat{\boldsymbol{\theta}}_1} = \boldsymbol{0}_{k \times |\mathcal{T}|}$ to ensure the invertibility

of $\boldsymbol{L}$ and that the constraints are met. This leads to

$$\sum_{j=1}^{r}\left(l_{t_j}\left(\hat{\boldsymbol{\theta}}_j\right)-l_{t_j}(\boldsymbol{u})\right) \leq \eta \sum_{j=1}^{r} \boldsymbol{\gamma}_5(R)|\mathcal{T}| + {}^1\!/\!_\eta \left(qd+\boldsymbol{\gamma}_5(R)|\mathcal{T}|-\mu_2 k\zeta\right) .$$

Choosing $\forall t_j\ \eta_{t_j}=\eta={}^1\!/\!_{\sqrt{R}}$, we acquire sublinear regret, finalizing the statement of Theorem 1:

$$\begin{aligned}\sum_{j=1}^{r}\left(l_{t_j}\left(\hat{\boldsymbol{\theta}}_j\right)-l_{t_j}(\boldsymbol{u})\right) &\leq {}^1\!/\!_{\sqrt{R}}\boldsymbol{\gamma}_5(R)|\mathcal{T}|R \\ &\quad + \sqrt{R}\left(qd+\boldsymbol{\gamma}_5(R)|\mathcal{T}|-\mu_2 k\zeta\right) \\ &\leq \sqrt{R}\Big(\boldsymbol{\gamma}_5(R)|\mathcal{T}|+qd\boldsymbol{\gamma}_5(R)|\mathcal{T}|-\mu_2 k\zeta\Big) \\ &\leq \mathcal{O}\left(\sqrt{R}\right) . \end{aligned}$$

□

6. Experimental Validation

To validate the empirical performance of our method, we applied our safe online PG algorithm to learn multiple consecutive control tasks on three dynamical systems (Figure 1). To generate multiple tasks, we varied the parameterization of each system, yielding a set of control tasks from each domain with varying dynamics. The optimal control policies for these systems vary widely with only minor changes in the system parameters, providing substantial diversity among the tasks within a single domain.

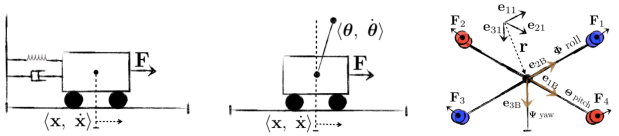


Figure 1. Dynamical systems used in the experiments: *a)* simple mass system (left), *b)* cart-pole (middle), and *c)* quadrotor unmanned aerial vehicle (right).

Simple Mass Spring Damper: The simple mass (SM) system is characterized by three parameters: the spring constant k in N/m, the damping constant d in Ns/m and the mass m in kg. The system's state is given by the position $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ of the mass, which varies according to a linear force $\boldsymbol{F}$. The goal is to train a policy for controlling the mass in a specific state $\boldsymbol{g}_{\text{ref}} = \langle \boldsymbol{x}_{\text{ref}}, \dot{\boldsymbol{x}}_{\text{ref}} \rangle$.

Cart Pole: The cart-pole (CP) has been used extensively as a benchmark for evaluating RL methods (Busoniu et al., 2010). CP dynamics are characterized by the cart's mass m_c in kg, the pole's mass m_p in kg, the pole's length in meters, and a damping parameter d in Ns/m. The state is given by the cart's position $\boldsymbol{x}$ and velocity $\dot{\boldsymbol{x}}$, as well as the pole's angle $\boldsymbol{\theta}$ and angular velocity $\dot{\boldsymbol{\theta}}$. The goal is to train a policy that controls the pole in an upright position.

6.1. Experimental Protocol

We generated 10 tasks for each domain by varying the system parameters to ensure a variety of tasks with diverse optimal policies, including those with highly chaotic dynamics that are difficult to control. We ran each experiment for a total of R rounds, varying from 150 for the simple mass to $10,000$ for the quadrotor to train $\boldsymbol{L}$ and $\boldsymbol{S}$, as well as for updating the PG-ELLA and PG models. At each round j, the learner observed a task t_j through 50 trajectories of 150 steps and updated $\boldsymbol{L}$ and $\boldsymbol{s}_{t_j}$. The dimensionality k of the latent space was chosen independently for each domain via cross-validation over 3 tasks, and the learning step size for each task domain was determined by a line search after gathering 10 trajectories of length 150. We used eNAC, a standard PG algorithm, as the base learner.

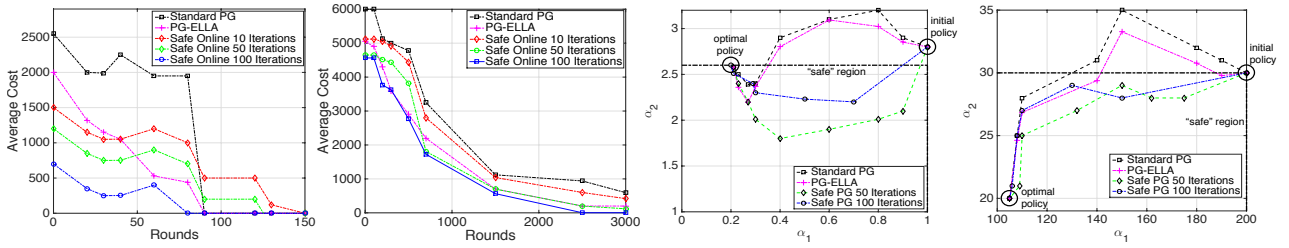
We compared our approach to both standard PG (i.e., eNAC) and PG-ELLA (Bou Ammar et al., 2014), examining both the constrained and unconstrained variants of our algorithm. We also varied the number of iterations in our alternating optimization from 10 to 100 to evaluate the effect of these inner iterations on the performance, as shown in Figures 2 and 3. For the two MTL algorithms (our approach and PG-ELLA), the policy parameters for each task t_j were initialized using the learned basis (i.e., $\boldsymbol{\alpha}_{t_j} = \boldsymbol{L}\boldsymbol{s}_{t_j}$). We configured PG-ELLA as described by Bou Ammar et al. (2014), ensuring a fair comparison. For the standard PG learner, we provided additional trajectories in order to ensure a fair comparison, as described below.

For the experiments with policy constraints, we generated a set of constraints $(\boldsymbol{A}_t, \boldsymbol{b}_t)$ for each task that restricted the policy parameters to pre-specified "safe" regions, as shown in Figures 2(c) and 2(d). We also tested different values for the constraints on $\boldsymbol{L}$, varying p and q between 0.1 to 10; our approach showed robustness against this broad range, yielding similar average cost performance.

6.2. Results on Benchmark Systems

Figure 2 reports our results on the benchmark simple mass and cart-pole systems. Figures 2(a) and 2(b) depicts the performance of the learned policy in a lifelong learning setting over consecutive unconstrained tasks, averaged over all 10 systems over 100 different initial conditions. These results demonstrate that our approach is capable of outperforming both standard PG (which was provided with 50 *additional* trajectories each iteration to ensure a more fair comparison) and PG-ELLA, both in terms of initial performance and learning speed. These figures also show that the performance of our method increases as it is given more alternating iterations per-round for fitting $\boldsymbol{L}$ and $\boldsymbol{S}$.

We evaluated the ability of these methods to respect safety constraints, as shown in Figures 2(c) and 2(d). The thicker black lines in each figure depict the allowable "safe" region of the policy space. To enable online learning per-task, the same task t_j was observed on each round and the shared basis $\boldsymbol{L}$ and coefficients $\boldsymbol{s}_{t_j}$ were updated using alternating optimization. We then plotted the change in the policy pa-



(a) Simple Mass (b) Cart Pole (c) Trajectory Simple Mass (d) Trajectory Cart Pole

Figure 2. Results on benchmark simple mass and cart-pole systems. Figures (a) and (b) depict performance in lifelong learning scenarios over consecutive unconstrained tasks, showing that our approach outperforms standard PG and PG-ELLA. Figures (c) and (d) examine the ability of these method to abide by safety constraints on sample constrained tasks, depicting two dimensions of the policy space (α_1 vs α_2) and demonstrating that our approach abides by the constraints (the dashed black region).

rameter vectors per iterations (i.e., $\boldsymbol{\alpha}_{t_j} = \boldsymbol{L}\boldsymbol{s}_{t_j}$) for each method, demonstrating that our approach abides by the safety constraints, while standard PG and PG-ELLA can violate them (since they only solve an unconstrained optimization problem). In addition, these figures show that increasing the number of alternating iterations in our method causes it to take a more direct path to the optimal solution.

6.3. Application to Quadrotor Control

We also applied our approach to the more challenging domain of quadrotor control. The dynamics of the quadrotor system (Figure 1) are influenced by inertial constants around $\boldsymbol{e}_{1,B}$, $\boldsymbol{e}_{2,B}$, and $\boldsymbol{e}_{3,B}$, thrust factors influencing how the rotor's speed affects the overall variation of the system's state, and the lengths of the rods supporting the rotors. Although the overall state of the system can be described by a 12-dimensional vector, we focus on stability and so consider only six of these state-variables. The quadrotor system has a high-dimensional action space, where the goal is to control the four rotational velocities $\{w_i\}_{i=1}^4$ of the rotors to stabilize the system. To ensure realistic dynamics, we used the simulated model described by (Bouabdallah, 2007; Voos & Bou Ammar, 2010), which has been verified and used in the control of physical quadrotors.

We generated 10 different quadrotor systems by varying the inertia around the x, y and z-axes. We used a linear quadratic regulator, as described by Bouabdallah (2007), to initialize the policies in both the learning and testing phases. We followed a similar experimental procedure to that discussed above to update the models.

Figure 3 shows the performance of the unconstrained solution as compared to standard PG and PG-ELLA. Again, our approach clearly outperforms standard PG and PG-ELLA in both the initial performance and learning speed. We also evaluated constrained tasks in a similar manner, again showing that our approach is capable of respecting constraints. Since the policy space is higher dimensional, we cannot visualize it as well as the benchmark systems, and so instead report the number of iterations it takes our approach

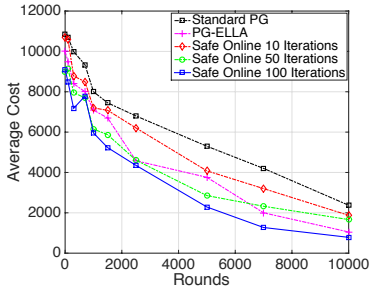


Figure 3. Performance on quadrotor control.

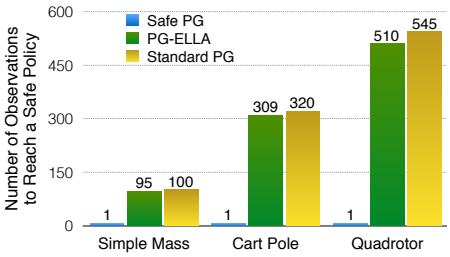


Figure 4. Average number of task observations before acquiring policy parameters that abide by the constraints, showing that our approach immediately projects policies to safe regions.

to project the policy into the safe region. Figure 4 shows that our approach requires only one observation of the task to acquire safe policies, which is substantially lower then standard PG or PG-ELLA (e.g., which require 545 and 510 observations, respectively, in the quadrotor scenario).

7. Conclusion

We described the first lifelong PG learner that provides sublinear regret $\mathcal{O}(\sqrt{R})$ with R total rounds. In addition, our approach supports safety constraints on the learned policy, which are essential for robust learning in real applications. Our framework formalizes lifelong learning as online MTL with limited resources, and enables safe transfer by sharing policy parameters through a latent knowledge base that is efficiently updated over time.

Acknowledgements

This research was supported by ONR grant #N00014-11-1-0139 and AFRL grant #FA8750-14-1-0069. We thank the anonymous reviewers for their helpful feedback.

References

Yasin Abbasi-Yadkori, Peter Bartlett, Varun Kanade, Yevgeny Seldin, & Csaba Szepesvári. Online learning in Markov decision processes with adversarially chosen transition probability distributions. *Advances in Neural Information Processing Systems* 26, 2013.

Haitham Bou Ammar, Karl Tuyls, Matthew E. Taylor, Kurt Driessen, & Gerhard Weiss. Reinforcement learning transfer via sparse coding. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems* (AAMAS), 2012.

Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, & Matthew Taylor. Online multi-task learning for policy gradient methods. In *Proceedings of the 31st International Conference on Machine Learning* (ICML), 2014.

Samir Bouabdallah. *Design and Control of Quadrotors with Application to Autonomous Flying*. PhD Thesis, École polytechnique fédérale de Lausanne, 2007.

Stephen Boyd & Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, 2004.

Lucian Busoniu, Robert Babuska, Bart De Schutter, & Damien Ernst. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, Boca Raton, FL, 2010.

Eliseo Ferrante, Alessandro Lazaric, & Marcello Restelli. Transfer of task representation in reinforcement learning using policy-based proto-value functions. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems* (AAMAS), 2008.

Mohammad Gheshlaghi Azar, Alessandro Lazaric, & Emma Brunskill. Sequential transfer in multi-armed bandit with finite set of models. *Advances in Neural Information Processing Systems* 26, 2013.

Roger A. Horn & Roy Mathias. Cauchy-Schwarz inequalities associated with positive semidefinite matrices. *Linear Algebra and its Applications* 142:63–82, 1990.

Jens Kober & Jan Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84(1–2):171–203, 2011.

Abhishek Kumar & Hal Daumé III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Conference on Machine Learning* (ICML), 2012.

Alessandro Lazaric. Transfer in reinforcement learning: a framework and a survey. In M. Wiering & M. van Otterlo, editors, *Reinforcement Learning: State of the Art*. Springer, 2011.

Jan Peters & Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 2008a.

Jan Peters & Stefan Schaal. Natural Actor-Critic. *Neurocomputing* 71, 2008b.

Paul Ruvolo & Eric Eaton. ELLA: An Efficient Lifelong Learning Algorithm. In *Proceedings of the 30th International Conference on Machine Learning* (ICML), 2013.

Richard S. Sutton & Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, 1998.

Richard S. Sutton, David Mcallester, Satinder Singh, & Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems* 12, 2000.

Matthew E. Taylor & Peter Stone. Transfer learning for reinforcement learning domains: a survey. *Journal of Machine Learning Research*, 10:1633–1685, 2009.

Sebastian Thrun & Joseph O'Sullivan. Discovering structure in multiple learning tasks: the TC algorithm. In *Proceedings of the 13th International Conference on Machine Learning* (ICML), 1996a.

Sebastian Thrun & Joseph O'Sullivan. Learning more from less data: experiments in lifelong learning. *Seminar Digest*, 1996b.

Holger Voos & Haitham Bou Ammar. Nonlinear tracking and landing controller for quadrotor aerial robots. In *Proceedings of the IEEE Multi-Conference on Systems and Control*, 2010.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8(3–4):229–256, 1992.

Aaron Wilson, Alan Fern, Soumya Ray, & Prasad Tadepalli. Multi-task reinforcement learning: a hierarchical Bayesian approach. In *Proceedings of the 24th International Conference on Machine Learning* (ICML), 2007.

Jian Zhang, Zoubin Ghahramani, & Yiming Yang. Flexible latent variable models for multi-task learning. *Machine Learning*, 73(3):221–242, 2008.