Safe Exploration in Gaussian Policy Gradient

Matteo Papini
Politecnico di Milano
matteo.papini@polimi.it

Andrea Battistello
Politecnico di Milano
andrea.battistello@mail.polimi.it

Marcello Restelli
Politecnico di Milano
marcello.restelli@polimi.it

Abstract

In many Reinforcement Learning (RL) applications, the goal is to find an optimal deterministic controller. However, most RL algorithms require the behavioral policy to be stochastic in order to perform a sufficient amount of exploration. Adjusting the level of stochasticity during the learning process is not trivial, as it is difficult to assess whether the costs of the exploration will be repaid in the long run, and the risks of instability and unsafe behavior are high. We study this problem in the context of policy gradients (PG) with Gaussian policies. Using tools from the safe PG literature, we devise a way to learn the policy variance that captures the effects this parameter has on the learning speed and on the quality of the final solution. Furthermore, we provide a way to update the policy variance safely. To do so, we generalize the existing bounds on performance improvement to the adaptive-variance case. We propose to combine these two techniques, thus obtaining the Safely-Exploring Policy Gradient (SEPG) approach. We evaluate the proposed methods on simulated continuous control tasks.

1 Introduction

Reinforcement learning (RL) [60] is an approach to adaptive intelligence that employs a reward signal to train an autonomous agent on a general task through direct interaction with the environment. The results recently achieved by RL in games [39, 57, 44, 67] are astounding. However, in order to apply RL to real-world scenarios (e.g., robotics, autonomous driving), we have to tackle further challenges. Unlike games, problems involving physical systems are more naturally modeled as continuous control tasks. For this reason, we will focus on policy gradient (PG) [61, 17], an RL technique that employs stochastic gradient ascent to optimize parametric controllers. PG is particularly suitable for continuous tasks due to its robustness to noise, convergence properties, and versatility in policy design [50]. Another perk of games is that they are easily simulated. Simulations require a reliable model of the environment, which is not always available. Learning on a real physical system (e.g., a robot) is often unavoidable and requires to deal with the *safe exploration* problem [24, 41, 19, 5, 65].

Although we normally look for a deterministic controller, the stochastic nature of the policy during the learning process is necessary to maintain a sufficient level of exploration. Exploration can be defined, in very general terms, as the execution of unprecedented behaviors in order to gather useful information. The transition from a stochastic policy to a deterministic one is non-trivial, as it falls within the exploration-exploitation dilemma. If exploration is abandoned too soon, or too fast, the agent may never know all the relevant aspects of the task and get stuck in suboptimal behavior. If the transition to deterministic behavior is delayed too much, the learning process may become unnecessarily long and expensive. Intuition suggests to adapt the level of exploration as the learning process progresses, but finding the right schedule is challenging. This kind of problem has been thoroughly studied in the Multi-Armed Bandit (MAB) literature [11, 37]. Exploration has a long history also in RL, mostly limited to the tabular setting [32, 9, 58, 27, 36, 15, 16, 28, 43, 46], with extensions to continuous states [45, 33, 8]. So far, exploration methods in PG [26, 22, 23, 13, 42, 56] have focused on accelerating the learning process, with less attention to the risks and costs of exploration. The need for *safe* exploration arises whenever the actions the agent performs *while learning* may have concrete consequences (e.g., monetary losses, harm to people and equipment). In the RL literature, safety has been modeled in several different ways [20]. In this work, we assume all sources of risk are encoded in the reward signal presented to the agent. This removes the necessity of additional "safety signals" (e.g., human advice [1]), although designing a sufficiently informative reward function may not be trivial. Safety can then be induced by constraining the reward optimization process [29, 40, 62, 38]. A common constraint [21, 63, 35] is that the expected return must not fall below a given threshold[1]. As noted by [20], this can be seen as an *indirect* modification of the exploration process, in contrast to more direct approaches based on prior knowledge [e.g., 19]. In the case of PG, safety can be enforced through conservative meta-parameter schedules [30, 51, 52, 47, 48].

Combining safety with adaptive exploration requires to find a good balance between these competing forces [14]. In this paper, we study an aspect of safe exploration, that is regulating the amount of stochasticity of the policy [2], in the context of policy gradients with Gaussian policies. In this case, the level of stochasticity can be controlled directly through the policy-variance meta-parameters. Our contribution is two-fold: we propose MEPG (Meta-Exploring Policy Gradients), a variant of PG that for the unconstrained setting (i.e., when the safety is not an issue) learns the policy variance via meta-gradient ascent on a far-sighted exploration objective; we also propose SEPG (Safely Exploring Policy Gradients), an adaptive-step-size version of MEPG that is safe while actively pursuing explorative behavior. The paper is structured as follows: in Section 2, we provide an essential background on PG and review the existing safety guarantees for this framework. In Section 3, we present our novel exploration objective and the MEPG algorithm. In Section 4, we extend existing improvement guarantees for Gaussian policies to the adaptive-variance case, providing safe exact-gradient updates. In section 5, we generalize these results to the more realistic stochastic-gradient case, and present the SEPG algorithm. Finally, in Section 6, we evaluate the proposed algorithms on simulated control tasks. Proofs of all the formal statements are given in Appendix A and C.

2 Preliminaries

In this section, we provide an essential background on policy gradient methods, including existing safety guarantees.

2.1 Policy Gradient Fundamentals

A continuous Markov Decision Process (MDP) [53] $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho \rangle$ is defined by a continuous state space $\mathcal{S} \subseteq \mathbb{R}^d$; a continuous action space $\mathcal{A}$; a Markovian transition kernel $\mathcal{P}$, where $\mathcal{P}(s'|s,a)$ is the transition density from state s to s' under action a; a reward function $\mathcal{R}$, where $\mathcal{R}(s,a) \in [-R_{\max}, R_{\max}]$ is the reward for state-action pair (s,a) and $R_{\max}$ is the maximum absolute-value reward; a discount factor $\gamma \in [0,1)$; and an initial-state distribution ρ on $\mathcal{S}$. An agent's behavior is modeled as a policy π, where $\pi(\cdot|s)$ is the density function over $\mathcal{A}$ in state s. We study episodic MDPs with indefinite horizon. In practice, we consider episodes of length H, the effective horizon of the task. A trajectory τ is a sequence of states and actions $(s_0, a_0, s_1, a_1, \dots, s_{H-1}, a_{H-1})$ observed by following a stationary policy, where $s_0 \sim \rho$ and $s_{h+1} \sim \mathcal{P}(\cdot|s_h, a_h)$. The policy induces a measure p_π over trajectories. We denote with $\mathcal{R}(\tau)$ the total discounted reward provided by trajectory τ: $\mathcal{R}(\tau) = \sum_{h=0}^{H-1} \gamma^h \mathcal{R}(s_h, a_h)$. Policies can be ranked based on their expected total reward $J(\pi) = \mathbb{E}_{\tau \sim p_\pi}[\mathcal{R}(\tau)]$. Solving the MDP means finding an optimal policy $\pi^* \in \arg\max_\pi \{J(\pi)\}$.

Policy gradient (PG) methods restrict this optimization problem to a class of parametric policies $\Pi_\Theta = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^m\}$, so that $\pi_{\boldsymbol{\theta}}$ is differentiable w.r.t. $\boldsymbol{\theta}$. We denote the performance of a parametric policy $\pi_{\boldsymbol{\theta}}$ with $J(\boldsymbol{\theta})$. A locally optimal policy can be found via gradient ascent on the

[1]This kind of constraint is also used in *conservative bandits* [69], which model the problem of a company exploring new strategies while maintaining its revenue above a fixed threshold.

[2]Our analysis is restricted to *undirected exploration*. Efficient directed exploration in continuous-action MDPs is still largely an open problem [12].

performance measure:

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t + \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t), \qquad \text{where} \quad \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau) \mathcal{R}(\tau) \right], \tag{1}$$

t denotes the current iteration, $p_{\boldsymbol{\theta}}$ is short for $p_{\pi_{\boldsymbol{\theta}}}$, and α is a step size. This policy search technique is known as policy gradient (PG) [61, 49]. In practice, $\nabla_{\boldsymbol{\theta}} J$ is not available, but can be estimated from a batch of trajectories $\mathcal{D}_N = \{\tau_1, \dots, \tau_N\}$. The GPOMDP [7] algorithm (a refinement of REINFORCE [68]) provides an unbiased estimator:

$$\widehat{\nabla}^N_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{h=0}^{H-1} \left(\sum_{i=0}^{h} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_i^n | s_i^n) \right) \left(\gamma^h \mathcal{R}(s_h^n, a_h^n) - b \right), \tag{2}$$

where b is a baseline used to reduce variance. Any baseline that does not depend on actions preserves the unbiasedness of the estimator.[3] We employ the variance-minimizing baselines provided by [49].

A widely used [18] policy class is the Gaussian[4]:

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{1}{\sqrt{2\pi}\sigma_\omega} \exp\left\{ -\frac{1}{2} \left(\frac{a - \mu_{\boldsymbol{\upsilon}}(s)}{\sigma_\omega} \right)^2 \right\}, \tag{3}$$

also denoted with $\mathcal{N}(a|\mu_{\boldsymbol{\upsilon}}(s), \sigma_\omega^2)$, where the action space is $\mathcal{A} = \mathbb{R}$, $\mu_{\boldsymbol{\upsilon}}$ is the state-dependent mean and $\sigma_\omega^2 > 0$ is the variance (σ_ω is the standard deviation). The policy parameters consist of a vector of mean parameters $\boldsymbol{\upsilon} \in \Upsilon \subseteq \mathbb{R}^m$ and a variance parameter $\omega \in \Omega \subseteq \mathbb{R}$, i.e., $\boldsymbol{\theta} \equiv [\boldsymbol{\upsilon}^T | \omega]^T$ and $\Theta \equiv \Upsilon \times \Omega \subseteq \mathbb{R}^{m+1}$. We focus on the following, common [54] parametrization:

$$\mu_{\boldsymbol{\upsilon}}(s) = \boldsymbol{\upsilon}^T \boldsymbol{\phi}(s), \qquad \sigma_\omega = e^\omega, \tag{4}$$

where $\boldsymbol{\phi}(\cdot)$ is a vector of m state-features[5]. We also assume that the state features are bounded in Euclidean norm, i.e., $\sup_{s \in \mathcal{S}} \|\boldsymbol{\phi}(s)\| \leqslant \varphi$, and both Υ and Ω are convex sets.

2.2 Safe Policy Gradients

As stated in Section 1, we are concerned with safety in the sense of guaranteeing a minimum performance threshold within the learning process. A convenient way to model this problem is through an *improvement constraint* [63]:

Definition 2.1. Given a parametric policy $\pi_{\boldsymbol{\theta}}$ with current parameter $\boldsymbol{\theta}_t$, we say that update $\Delta\boldsymbol{\theta} \in \mathbb{R}^{m+1}$ is safe w.r.t. requirement $C_t \in \mathbb{R}$ if:

$$J(\boldsymbol{\theta}_{t+1}) - J(\boldsymbol{\theta}_t) \geqslant C_t, \tag{5}$$

where $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \Delta\boldsymbol{\theta}$.

Given a performance threshold $J_{\min}$, we can ensure $J(\boldsymbol{\theta}_{t+1}) \geqslant J_{\min}$ by setting $C_t = J_{\min} - J(\boldsymbol{\theta}_t)$. In general, we talk of a *required performance improvement* when requirement C_t is non-negative, otherwise of a *bounded worsening*. The case $C_t \equiv 0$ corresponds to the well-studied *monotonic improvement* constraint [30, 51, 47]. Recent work [48] provides improvement guarantees for a general family of policies. Given positive constants ψ, κ and ξ, a policy class Π_Θ is called (ψ, κ, ξ)-smoothing if:

$$\sup_{s \in \mathcal{S}} \mathbb{E}_{a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \Big[\|\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\| \Big] \leqslant \psi, \qquad \sup_{s \in \mathcal{S}} \mathbb{E}_{a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \Big[\|\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\|^2 \Big] \leqslant \kappa,$$

$$\sup_{s \in \mathcal{S}} \mathbb{E}_{a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \Big[\|\nabla\nabla^T \log \pi_{\boldsymbol{\theta}}(a|s)\| \Big] \leqslant \xi, \tag{6}$$

[3]Action-dependent baselines are also possible. See [64] for a discussion.

[4]We consider scalar actions for simplicity. Multi-dimensional actions are discussed in Appendix E, together with heteroskedastic exploration.

[5]This is a shallow policy since we assume to have the features already available, as opposed to a deep policy, where the features are learned together with $\boldsymbol{\upsilon}$ in an end-to-end fashion.

for all $\boldsymbol{\theta} \in \Theta$, where $\|\cdot\|$ denotes the Euclidean norm for vectors and the spectral norm for matrices. In particular, Gaussian policies *with fixed standard deviation* (constant ω) are (ψ, κ, ξ)-smoothing with the following constants [48]:

$$\psi = \frac{2\varphi}{\sqrt{2\pi}\sigma_\omega}, \qquad \kappa = \xi = \frac{\varphi^2}{\sigma_\omega^2}, \tag{7}$$

where φ is the Euclidean-norm bound on state features. For a (ψ, κ, ξ)-smoothing policy, the performance improvement yielded by a policy gradient update can be lower bounded by a function of the step size α as follows [Theorem 9 from 48]:

$$J(\boldsymbol{\theta}_{t+1}) - J(\boldsymbol{\theta}_t) \geqslant \alpha \left\|\nabla J(\boldsymbol{\theta}_t)\right\|^2 - \alpha^2 \frac{L}{2} \left\|\Delta\boldsymbol{\theta}\right\|^2, \tag{8}$$

where $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\nabla J(\boldsymbol{\theta}_t)$ and $L = \frac{R_{\max}}{(1-\gamma)^2}\left(\frac{2\gamma\psi^2}{1-\gamma} + \kappa + \xi\right)$. This allows, given an improvement constraint C_t as in (5), to select a safe step size. In this paper, whenever multiple choices of the step size satisfy the constraint of interest, we decide to employ the *largest* safe step size. This is meant to yield faster convergence (see Section 6). In the fixed-variance Gaussian case, we can obtain a safe step size for the mean-parameter update [adaptation of Corollary 10 from 48]:

Lemma 2.1. *Let Π_Υ be the class of Gaussian policies parametrized as in* (4)*, but with* fixed variance parameter ω. *Let $\boldsymbol{v}_t \in \mathbb{R}^m$ and $\boldsymbol{v}_{t+1} = \boldsymbol{v}_t + \alpha_t\nabla_{\boldsymbol{v}} J(\boldsymbol{v}_t, \omega)$. For any $C_t \leqslant C_t^*$, the largest step size guaranteeing $J(\boldsymbol{v}_{t+1}, \omega) - J(\boldsymbol{v}_t, \omega) \geqslant C_t$ is:*

$$\overline{\alpha}_t := \frac{\sigma_\omega^2}{F}\left(1 + \sqrt{1 - C_t/C_t^*}\right), \tag{9}$$

where $F = \frac{2\varphi^2 R_{\max}}{(1-\gamma)^2}\left(1 + \frac{2\gamma}{\pi(1-\gamma)}\right)$ and $C_t^* = \frac{\sigma_\omega^2\|\nabla_{\boldsymbol{v}} J(\boldsymbol{v}_t,\omega)\|^2}{2F}$.

We have highlighted the role of the policy standard deviation. We can see how a larger σ_ω allows to take larger steps. Moreover, it increases the maximum improvement guarantee that one can ask for. In fact, both $\overline{\alpha}_t$ and C_t^* are $\mathcal{O}(\sigma_\omega^2)$. This is due to the smoothing effect of the policy variance on the optimization landscape, in accordance with the empirical analysis from [2]. In [48], the step size maximizing the immediate guaranteed improvement is always selected. This corresponds to setting $C_t \equiv C_t^*$ in (9). High probability variants of Lemma 2.1 are available for the more realistic stochastic-gradient case [48].

3 Adaptive Exploration

In this section, we use some insights from the safe PG literature to devise a general-purpose (i.e., without safety guarantees) approach to adapt the standard deviation of a Gaussian policy during the learning process. The safe counterpart is presented in the next section.

Consider a Gaussian policy $\pi_{\boldsymbol{v},\omega}(a|s) = \mathcal{N}(a|\mu_{\boldsymbol{v}}(s), \sigma_\omega)$, parametrized as in (4). As mentioned above, it is common to learn the policy variance parameter via gradient ascent just like any other parameter, i.e., $\omega_{t+1} \leftarrow \omega_t + \beta_t\nabla_\omega J(\boldsymbol{v}_t, \omega_t)$. However, the effects of σ on the optimization landscape, exposed by Lemma 2.1, suggest to treat it with particular care, both to exploit its potential and to avoid its possible risks. In fact, adjusting the policy variance with policy gradient tends to degenerate too early into quasi-deterministic policies, getting stuck in local optima or even causing divergence issues (see Section 6). We use our understanding of the special nature of this parameter to modify GPOMDP in two ways. First of all, we make the step size dependent on the policy variance, like the safe step size from Lemma 2.1. In particular, we use the following:

$$\alpha_t = \alpha\sigma_{\omega_t}^2 / \|\nabla_{\boldsymbol{v}} J(\boldsymbol{v}_t, \omega_t)\|, \tag{10}$$

to update the mean parameters $\boldsymbol{v}$, where $\alpha > 0$ is a meta-parameter. This has both the effect of reducing the step size when a small σ makes the optimization landscape less smooth, preventing oscillations, and increasing it when a large σ allows it to do so, increasing the learning speed. This is not entirely unheard of, as it is exactly what a natural gradient [31, 4] would do in a pure Gaussian setting ($1/\sigma^2$ is the Fisher information w.r.t. the mean parameters of a Gaussian distribution). We also divide the step size by the norm of the gradient. This is a common normalization technique [50], and

Algorithm 1 MEPG

```
1: Input: Initial parameters $\boldsymbol{\upsilon}_0$ and $\omega_0$, step size $\alpha > 0$, meta step size $\eta > 0$, batch size $N$
2: for $t = 1, 2, \dots$ do
3:     Estimate $\widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)$, $\widehat{\nabla}_\omega J(\boldsymbol{\upsilon}_t, \omega_t)$ and $\widehat{\nabla}_\omega \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\|$
4:     $\widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) = \widehat{\nabla}_\omega J(\boldsymbol{\upsilon}_t, \omega_t) + \alpha e^{2\omega_t} \left( 2 \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\| + \widehat{\nabla}_\omega \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\| \right)$
5:     $\omega_{t+1} \leftarrow \omega_t + \eta \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) / \left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) \right\|$
6:     $\boldsymbol{\upsilon}_{t+1} \leftarrow \boldsymbol{\upsilon}_t + \alpha e^{2\omega_t} \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) / \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\|$
7: end for
```

is further motivated by the results of Section 5 on stochastic gradient updates. Then, we treat ω as a meta-parameter and we learn it in a meta-gradient fashion [59, 55, 66, 70]. Specifically, we employ a more far-sighted learning objective to avoid premature convergence to deterministic behavior. To do so, we look at the target performance at one step in the future:

$$J\left(\boldsymbol{\upsilon}_t + \alpha\sigma^2_{\omega_t} \frac{\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)}{\left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\|},\, \omega_t\right) \simeq J(\boldsymbol{\upsilon}_t, \omega_t) + \alpha\sigma^2_{\omega_t} \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\| := \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t), \tag{11}$$

where we performed a first-order approximation. The gradient of $\mathcal{L}$ w.r.t. ω is:

$$\nabla_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) = \nabla_\omega J(\boldsymbol{\upsilon}_t, \omega_t) + 2\alpha\sigma^2_{\omega_t} \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\| + \alpha\sigma^2_{\omega_t} \nabla_\omega \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\|. \tag{12}$$

The first term of the sum is the usual policy gradient w.r.t. ω, and measures the direct effect of policy stochasticity on performance. The role of the second term is to increase the step size α_t, more so if the gradient w.r.t. $\boldsymbol{\upsilon}$ is large. The third term is meant to modify the policy variance to increase the gradient norm and can be seen as a way to escape local optima. The last two terms, together, account for the long-term effects of modifying the policy variance. We propose to update ω in the direction of the (normalized) meta-gradient $\nabla_\omega \mathcal{L}$ using a meta-step size $\eta > 0$:

$$\omega_{t+1} \leftarrow \omega_t + \eta \frac{\nabla_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t)}{\left\| \nabla_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) \right\|}. \tag{13}$$

In practice, exact gradients are not available. The policy gradient for the mean-parameter update can be estimated with GPOMDP (2). Computing $\widehat{\nabla}_\omega \mathcal{L}$ further requires estimating $\nabla_\omega \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\|$ (See Appendix B). The pseudocode for the resulting algorithm, called Meta-Exploring Policy Gradient (MEPG), is provided in Algorithm 1.

4 Safe Exploration

In this section, we extend the performance improvement guarantees reported in Section 2.2 to Gaussian policies with adaptive variance, and we use these theoretical results to devise a safe version of Algorithm 1. The improvement guarantees for fixed-variance Gaussian policies are based on the smoothing constants from [48], reported in (7). These depend (inversely) on σ^2_ω, hence are no longer constant once we allow ω to vary. In particular, they tend to infinity as the policy approaches determinism. Unfortunately, this is enough to invalidate the safety guarantees. A workaround would be to replace σ_ω with a lower bound, which can be imposed by constraining the parameter space Ω or by changing the parametrization. However, this would make the improvement bounds unnecessarily conservative, and would prevent the agent to converge to deterministic behavior. For these reasons, we instead propose to update the mean and variance parameters *alternately*First, we show that Gaussian policies are smoothing w.r.t. the variance parameter *alone*:

Lemma 4.1. *Let* Π_Ω *be the class of Gaussian policies parametrized as in* (4)*, but with* fixed mean parameter $\boldsymbol{\upsilon}$. Π_Ω *is* $\left(\frac{4}{\sqrt{2\pi e}}, 2, 2\right)$*-smoothing.*

This allows to devise a safe policy-gradient update for the variance parameters:

Theorem 4.2. *Let* Π_Ω *be the class of policies defined in Lemma 4.1. Let* $\omega_t \in \Omega$ *and* $\omega_{t+1} \leftarrow \omega_t + \beta_t \nabla_\omega J(\boldsymbol{\upsilon}, \omega_t)$*. For any* $C_t \leqslant C^*_t$*, the largest step-size satisfying* (5) *is:*

$$\overline{\beta}_t = \frac{1}{G}\left(1 + \sqrt{1 - C_t / C^*_t}\right), \tag{14}$$

Algorithm 2 SEPG

```
1: Input: Initial parameters $\boldsymbol{\upsilon}_0$ and $\omega_0$, batch size $N$, improvement thresholds $\{C_t\}_{t=1}^{\infty}$, confidence
   parameter $\delta$, discount factor $\gamma$, maximum reward $R_{\max}$, feature bound $\varphi$
2: for $t = 1, 2, \dots$ do
3:     Estimate $\widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)$,
4:     if $t$ is odd then
5:         $\boldsymbol{\upsilon}_{t+1} = \boldsymbol{\upsilon}_t + \widetilde{\alpha}_t \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)$        ▷ Safe step size $\widetilde{\alpha}_t$ from Equation (18)
6:     else
7:         estimate $\widehat{\nabla}_{\omega} J(\boldsymbol{\upsilon}_t, \omega_t)$ and $\widehat{\nabla}_{\omega} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\|$
8:         $\widehat{\nabla}_{\omega} \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) = \widehat{\nabla}_{\omega} J(\boldsymbol{\upsilon}_t, \omega_t) + \widetilde{\alpha}_t \left( 2 \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\| + \widehat{\nabla}_{\omega} \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t) \right\| \right)$
9:         $\omega_{t+1} = \omega_t + \widetilde{\eta}_t \nabla_{\omega} \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t)$        ▷ Safe meta step size $\widetilde{\eta}_t$ from Equation (19)
10:    end if
11: end for
```

where $C_t^* = \frac{\|\nabla_\omega J(\boldsymbol{\upsilon}, \omega_t)\|^2}{2G}$ *and* $G = \frac{4R_{\max}}{(1-\gamma)^2}\left(1 + \frac{4\gamma}{\pi e(1-\gamma)}\right)$.

Alternately updating the mean parameter as in Lemma 2.1 and the variance parameter as in Theorem 4.2 ensures $J(\boldsymbol{\upsilon}_{t+1}, \omega_{t+1}) - J(\boldsymbol{\upsilon}_t, \omega_t) \geqslant C_t$ for all t.

However, Theorem 4.2 still pertains *naïve* variance updates, which suffer from all the problems discussed in Section 3. The next question is how to optimize the surrogate exploratory objective $\mathcal{L}$ from 11 while satisfying the original constraint (5) on the performance objective J. The following Theorem provides a safe update for a smoothing policy in the direction of a generic update vector $\boldsymbol{x}_t$:

Theorem 4.3. *Let* Π_Θ *be a* (ψ, κ, ξ)*-smoothing policy class,* $\boldsymbol{\theta}_t \in \Theta$, *and* $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta_t \boldsymbol{x}_t$, *where* $\boldsymbol{x}_t \in \mathbb{R}^m$ *and* $\eta_t \in \mathbb{R}$ *is a (possibly negative) step size. Let* $\lambda_t := \frac{\langle \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t), \boldsymbol{x}_t \rangle}{\|\boldsymbol{x}_t\|}$ *be the scalar projection of* $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t)$ *onto* $\boldsymbol{x}_t$. *For any* $C_t \leqslant C_t^*$, *provided* $\lambda_t \neq 0$, *the largest step size guaranteeing* $J(\boldsymbol{\theta}_{t+1}) - J(\boldsymbol{\theta}_t) \geqslant C_t$ *is:*

$$\overline{\eta}_t = \frac{|\lambda_t|}{L\,\|\boldsymbol{x}_t\|}\left(sign(\lambda_t) + \sqrt{1 - C_t / C_t^*}\right), \tag{15}$$

where $C_t^* = \frac{\lambda_t^2}{2L}$ *and* $L = \frac{R_{\max}}{(1-\gamma)^2}\left(\frac{2\gamma\psi^2}{1-\gamma} + \kappa + \xi\right)$.

Note that a positive performance improvement up to C_t^* can always be guaranteed, even if the improvement direction $\nabla_{\boldsymbol{\theta}} J$ is not explicitly followed. However, when the scalar projection λ_t is negative, the largest safe step size is negative. This corresponds to the case in which maximizing the surrogate objective minimizes the original one. For positive values of C_t (required improvement), there may be no way to safely pursue the surrogate objective. In this case, a negative step size is prescribed to follow the direction of $\nabla_{\boldsymbol{\theta}} J$ instead[6]. We use the step size $\overline{\eta}_t$ from Theorem 4.3 to safely replace $\nabla_\omega J$ with the meta gradient $\nabla_\omega \mathcal{L}$ from (12) in the variance update:

$$\boldsymbol{\upsilon}_{t+1} \leftarrow \boldsymbol{\upsilon}_t + \overline{\alpha}_t \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t), \tag{16}$$

$$\omega_{t+2} \leftarrow \omega_{t+1} + \overline{\eta}_{t+1} \nabla_\omega \mathcal{L}(\boldsymbol{\upsilon}_{t+1}, \omega_{t+1}), \tag{17}$$

where $\omega_{t+1} \equiv \omega_t$ and $\boldsymbol{\upsilon}_{t+2} \equiv \boldsymbol{\upsilon}_{t+1}$, as the two set of parameters cannot be safely updated together.

5 Approximate Framework

In practice, exact gradients are not available and must be estimated from data. In this section, we show how to adapt the safe step sizes from Section 4 to take gradient estimation errors into account. Let $\widehat{\nabla}_{\boldsymbol{\upsilon}}^N J$, $\widehat{\nabla}_{\omega}^N J$ and $\widehat{\nabla}_{\omega}^N \mathcal{L}$ be unbiased estimators of $\nabla_{\boldsymbol{\upsilon}} J$, $\nabla_{\boldsymbol{\upsilon}} J$ and $\nabla_\omega \mathcal{L}$, respectively, each using a batch of N trajectories. As for MEPG, the first two can be GPOMDP estimators (2) and meta-gradient

[6] The special case when the two gradients are orthogonal ($\lambda_t = 0$) is discussed in Appendix A.

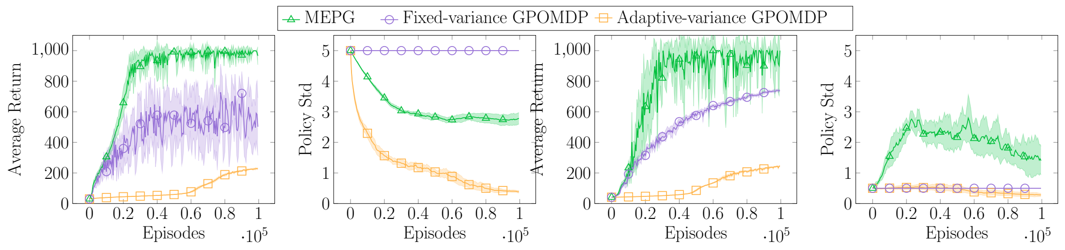

Figure 1: Average return (undiscounted) and policy standard deviation per episode of MEPG, adaptive-variance GPOMDP and fixed-variance GPOMDP on the continuous Cart-Pole task, starting from $\sigma = 5$ (left) and $\sigma = 0.5$ (right); averaged over 10 runs with 95% confidence intervals.

estimation is discussed in Appendix B. Following [48], we make the following assumption on the policy gradient estimators[7]:

Assumption 5.1. *For every $\delta \in (0,1)$ there exists a non-negative constant ϵ_δ such that, with probability at least $1-\delta$:*

$$\left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon},\omega) - \widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon},\omega)\right\| \leqslant \frac{\epsilon_\delta}{\sqrt{N}}, \qquad \left\|\nabla_{\omega} J(\boldsymbol{\upsilon},\omega) - \widehat{\nabla}^N_{\omega} J(\boldsymbol{\upsilon},\omega)\right\| \leqslant \frac{\epsilon_\delta}{\sqrt{N}},$$

for every $\boldsymbol{\upsilon} \in \Upsilon$, $\omega \in \Omega$ and $N \geqslant 1$.

Here ϵ_δ represents an upper bound on the gradient estimation error. This can be characterized using various statistical inequalities [47]. A possible one, based on ellipsoidal confidence regions, is described in Appendix D. Under Assumption 5.1, provided $N > \epsilon_\delta^2 / \left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t,\omega_t)\right\|^2$, the safe step size for the mean update can be adjusted as follows:

$$\widetilde{\alpha}_t = \frac{\sigma_\omega^2\left(\left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t,\omega_t)\right\| - \frac{\epsilon_\delta}{\sqrt{N}}\right)}{F\left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t,\omega_t)\right\|}\left(1+\sqrt{1-C_t/C_t^*}\right), \tag{18}$$

where $C_t^* = \frac{\sigma_{\omega_t}^2\left(\left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t,\omega_t)\right\| - \frac{\epsilon_\delta}{\sqrt{N}}\right)^2}{2F}$ and F is from Lemma 2.1. The constraint on the batch size ensures that the estimation error is not larger than the estimate itself. As expected, the maximum guaranteed improvement C_t^* is reduced w.r.t. the exact case.

Similarly, provided $N > \epsilon_\delta^2 / \widehat{\lambda}_t^2$, the safe step size for the variance update can be adjusted as follows:

$$\widetilde{\eta}_t = \frac{\left|\widehat{\lambda}_t\right| - \frac{\epsilon_\delta}{\sqrt{N}}}{G\left\|\widehat{\nabla}^N_{\omega} \mathcal{L}(\boldsymbol{\upsilon}_t,\omega_t)\right\|}\left(sign\left(\widehat{\lambda}_t\right)+\sqrt{1-C_t/C_t^*}\right), \tag{19}$$

where $\widehat{\lambda}_t$ is the scalar projection of $\widehat{\nabla}^N_{\omega} J(\boldsymbol{\upsilon}_t,\omega_t)$ onto $\widehat{\nabla}^N_{\omega} \mathcal{L}(\boldsymbol{\upsilon}_t,\omega_t)$, $C_t^* = \frac{\left(|\widehat{\lambda}_t| - \frac{\epsilon_\delta}{\sqrt{N}}\right)^2}{2G}$ and G is from Theorem 4.2. In this case, the constraint on the batch size ensures that the policy gradient estimation error is not larger than the projected surrogate gradient.

Under Assumption 5.1, these step sizes guarantee that our safety constraint (5) is satisfied with probability at least $1-\delta$ (see Appendix C for a formal treatment). We call the resulting algorithm Safely Exploring Policy Gradient (SEPG), and provide pseudocode in Algorithm 2. Note that $\widetilde{\alpha}$ already includes the $\sigma_\omega^2 / \left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t,\omega_t)\right\|$ term, which further motivates its addition in the non-safe context (10).

6 Experiments

In this section, we test the proposed methods on simulated continuous control tasks.

[7]We do not need a similar assumption on the meta-gradient estimator $\widehat{\nabla}_\omega \mathcal{L}$, since our improvement requirements are always on the performance J (see Appendix C).

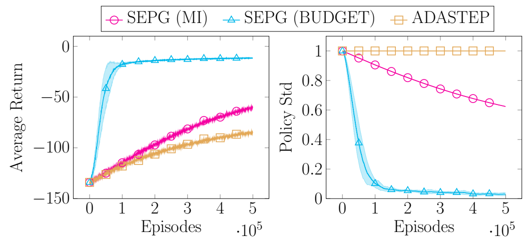

Figure 2: Average return (undiscounted) and standard deviation per episode of SEPG and ADASTEP on the LQG task, averaged over 10 runs with 95% confidence intervals.

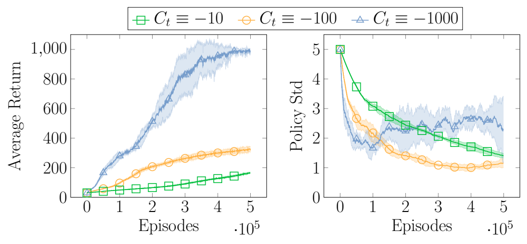

Figure 3: Average return (undiscounted) and standard deviation per episode of SEPG on the Cart-Pole task for different values of C_t, averaged over 5 runs with 95% confidence intervals.

MEPG We test MEPG on a continuous-action version of the Cart-Pole balancing task [6]. Figure 1 shows the performance (1000 is the maximum) and the policy standard deviation of MEPG, fixed-variance GPOMDP and adaptive-variance GPOMDP. For each algorithm, the best step sizes have been selected by grid search (see Appendix F and G for complete experimental details). Two very different initializations are considered for the standard deviation: $\sigma_{\omega_0} = 5$ (on the left) and $\sigma_{\omega_0} = 0.5$ (on the right). As shown by the behavior of fixed-variance GPOMDP, the former is too large to achieve optimal performance at convergence, while the latter is too small to properly explore the environment. As expected, adaptive-variance GPOMDP is too greedy and ends up always reducing the standard deviation. Besides preventing exploration, divergence issues force us to use a smaller step size, resulting in slower learning. Instead, MEPG is able to reduce the standard deviation from 5 and to increase it from 0.5, settling on an intermediate value in both cases. This allows both to learn faster and to achieve optimal performance, although non-negligible oscillations can be observed.

SEPG Figure 2 shows the performance and the policy standard deviation of SEPG on the one-dimensional LQG (Linear-Quadratic Gaussian regulator) task [49]. SEPG with a monotonic improvement constraint ($C_t \equiv 0$) is compared with the adaptive-step-size algorithm by [51] (ADASTEP in the figure). Starting from $\sigma_{\omega_0} = 1$, SEPG achieves higher returns by safely lowering it, while ADASTEP has no way to safely update this parameter. Both algorithms use $\delta = 0.2$ and a large batch size ($N = 500$). We also consider a looser constraint (BUDGET in the figure): that of never doing worse than the initial performance ($C_t = J(\boldsymbol{\theta}_0) - J(\boldsymbol{\theta}_t)$). As expected, this allows faster learning, leading to optimal performance within a reasonable time.

On the Cart-Pole task, MEPG showed an oscillatory behavior. Motivated by this fact, we run SEPG with a fixed, *negative* improvement threshold C_t. Recall that the meaning of such a constraint is to limit per-update performance worsening. Figure 3 shows the results for different values of the threshold, starting from $\sigma_{\omega_0} = 5$ and neglecting the gradient estimation error (i.e., by setting $\delta = 1$). Even under this simplifying assumption, only a very large value of C_t allows to reach optimal performance within a reasonable time[8]. Note how oscillations are reduced w.r.t. MEPG (Figure 1), and how policy standard deviation is first reduced and then *safely increased* again.

7 Conclusions

We have highlighted the special role of the standard deviation in PG with Gaussian policies. We have proposed a variant of GPOMDP for this setting, called Meta-Exploring Policy Gradient (MEPG), which is able to adapt this parameter in a more far-sighted way than plain gradient ascent. We have generalized the existing performance-improvement bounds for Gaussian policies to the adaptive-variance case. Combining these contributions, we have proposed SEPG, a Safely Exploring Policy Gradient algorithm. The experiments confirmed several intuitions provided by the theory. Future work should focus on improving the sample complexity of SEPG in order to tackle more challenging control tasks, either by tightening the improvement bounds or by employing heuristics. Aspects of exploration that are not captured by the mere selection of the variance parameter should also be enquired.

[8] The fact that this meta-parameter is out of scale is due to the over-conservativeness of the gradient Lipschitz constants. Replacing the theoretically sound upper bounds with estimates [3] could overcome this issue.

References

[1] Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[2] Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy in policy learning. *arXiv preprint arXiv:1811.11214*, 2018.

[3] Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. In *Advances in neural information processing systems*, pages 2675–2686, 2018.

[4] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[5] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[6] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Systems, Man, and Cybernetics*, 13(5):834–846, 1983.

[7] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15, 2001.

[8] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479, 2016.

[9] Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002.

[10] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[11] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[12] Sayak Ray Chowdhury and Aditya Gopalan. Online learning in kernelized markov decision processes. *CoRR*, abs/1805.08052, 2018.

[13] Kamil Ciosek and Shimon Whiteson. Expected policy gradients for reinforcement learning. *CoRR*, abs/1801.03326, 2018.

[14] Andrew Cohen, Lei Yu, and Robert Wright. Diverse exploration for fast and safe policy improvement. *arXiv preprint arXiv:1802.08331*, 2018.

[15] Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2818–2826, 2015.

[16] Christoph Dann, Tor Lattimore, and Emma Brunskill. Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 5713–5723, 2017.

[17] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.

[18] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1329–1338. JMLR.org, 2016.

[19] Javier García and Fernando Fernández. Safe exploration of state and action spaces in reinforcement learning. *J. Artif. Intell. Res.*, 45:515–564, 2012.

[20] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[21] Peter Geibel and Fritz Wysotzki. Risk-sensitive reinforcement learning applied to control under constraints. *J. Artif. Intell. Res.*, 24:81–108, 2005.

[22] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1352–1361, 2017.

[23] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, volume 80 of *JMLR Workshop and Conference Proceedings*, pages 1856–1865. JMLR.org, 2018.

[24] Alexander Hans, Daniel Schneegaß, Anton Maximilian Schäfer, and Steffen Udluft. Safe exploration for reinforcement learning. In *ESANN*, pages 143–148, 2008.

[25] Wolfgang Härdle and Léopold Simar. *Applied multivariate statistical analysis*, volume 22007. Springer, 2012.

[26] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, pages 1109–1117, 2016.

[27] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

[28] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, pages 4868–4878, 2018.

[29] Yoshinobu Kadota, Masami Kurano, and Masami Yasuda. Discounted markov decision processes with utility constraints. *Computers & Mathematics with Applications*, 51(2):279–284, 2006.

[30] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. 2002.

[31] Sham M Kakade. A natural policy gradient. In *Advances in neural information processing systems*, pages 1531–1538, 2002.

[32] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

[33] K. Lakshmanan, Ronald Ortner, and Daniil Ryabko. Improved regret bounds for undiscounted continuous reinforcement learning. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 524–532, 2015.

[34] Cornelius Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.

[35] Romain Laroche and Paul Trichelair. Safe policy improvement with baseline bootstrapping. *CoRR*, abs/1712.06924, 2017.

[36] Tor Lattimore and Marcus Hutter. Near-optimal pac bounds for discounted mdps. *Theoretical Computer Science*, 558:125–143, 2014.

[37] Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press (preprint), 2019.

[38] Jongmin Lee, Youngsoo Jang, Pascal Poupart, and Kee-Eung Kim. Constrained bayesian reinforcement learning via approximate linear programming. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 2088–2095. ijcai.org, 2017.

[39] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[40] Teodor Mihai Moldovan and Pieter Abbeel. Risk aversion in markov decision processes via near optimal chernoff bounds. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 3140–3148, 2012.

[41] Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. In *ICML*. icml.cc / Omnipress, 2012.

[42] Ofir Nachum, Mohammad Norouzi, George Tucker, and Dale Schuurmans. Smoothed action value functions for learning gaussian policies. In *International Conference on Machine Learning*, pages 3689–3697, 2018.

[43] Jungseul Ok, Alexandre Proutière, and Damianos Tranos. Exploration in structured reinforcement learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 8888–8896, 2018.

[44] OpenAI. Openai five. `https://blog.openai.com/openai-five/`, 2018.

[45] Ronald Ortner and Daniil Ryabko. Online regret bounds for undiscounted continuous reinforcement learning. In *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 2*, pages 1763–1771. Curran Associates Inc., 2012.

[46] Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3003–3011, 2013.

[47] Matteo Papini, Matteo Pirotta, and Marcello Restelli. Adaptive batch size for safe policy gradients. In *Advances in Neural Information Processing Systems*, pages 3591–3600, 2017.

[48] Matteo Papini, Matteo Pirotta, and Marcello Restelli. Smoothing policies and safe policy gradients. *arXiv preprint arXiv:1905.03231*, 2019.

[49] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

[50] Jan Peters, Sethu Vijayakumar, and Stefan Schaal. Natural actor-critic. In *European Conference on Machine Learning*, pages 280–291. Springer, 2005.

[51] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Adaptive step-size for policy gradient methods. In *Advances in Neural Information Processing Systems 26*, pages 1394–1402. 2013.

[52] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 100(2-3), 2015.

[53] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[54] Aravind Rajeswaran, Kendall Lowrey, Emanuel Todorov, and Sham M. Kakade. Towards generalization and simplicity in continuous control. In *NIPS*, pages 6553–6564, 2017.

[55] Nicol N Schraudolph. Local gain adaptation in stochastic gradient descent. 1999.

[56] Lior Shani, Yonathan Efroni, and Shie Mannor. Revisiting exploration-conscious reinforcement learning. *arXiv preprint arXiv:1812.05551*, 2018.

[57] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[58] Alexander L Strehl, Lihong Li, and Michael L Littman. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 10(Nov):2413–2444, 2009.

[59] Richard S Sutton. Adapting bias by gradient descent: An incremental version of delta-bar-delta. In *AAAI*, pages 171–176, 1992.

[60] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[61] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[62] Aviv Tamar, Dotan Di Castro, and Shie Mannor. Policy gradients with variance related risk criteria. In *Proceedings of the twenty-ninth international conference on machine learning*, pages 387–396, 2012.

[63] Philip S. Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence policy improvement. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2380–2388. JMLR.org, 2015.

[64] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. 2018.

[65] Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe exploration in finite markov decision processes with gaussian processes. In *Advances in Neural Information Processing Systems*, pages 4312–4320, 2016.

[66] Vivek Veeriah, Shangtong Zhang, and Richard S Sutton. Crossprop: Learning representations by stochastic meta-gradient descent in neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 445–459. Springer, 2017.

[67] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojciech M. Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Yuhuai Wu, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis, and David Silver. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. `https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/`, 2019.

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

[69] Yifan Wu, Roshan Shariff, Tor Lattimore, and Csaba Szepesvári. Conservative bandits. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1254–1262. JMLR.org, 2016.

[70] Zhongwen Xu, Hado P. van Hasselt, and David Silver. Meta-gradient reinforcement learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 2402–2413, 2018.

[71] Tingting Zhao, Hirotaka Hachiya, Gang Niu, and Masashi Sugiyama. Analysis and improvement of policy gradient estimation. In *NIPS*, pages 262–270, 2011.

Table of Supplementary Contents

A Proofs

In this section we provide proofs for all the formal statements made in the paper. Recall that $R_{\max}$ is the maximum absolute value reward, φ the Euclidean-norm bound on state features, γ the discount factor, and π with no subscript denotes the mathematical constant.

Lemma 2.1. *Let Π_Υ be the class of Gaussian policies parametrized as in* (4)*, but with* fixed variance parameter *ω. Let $\boldsymbol{\upsilon}_t \in \mathbb{R}^m$ and $\boldsymbol{\upsilon}_{t+1} = \boldsymbol{\upsilon}_t + \alpha_t \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega)$. For any $C_t \leqslant C_t^*$, the largest step size guaranteeing $J(\boldsymbol{\upsilon}_{t+1}, \omega) - J(\boldsymbol{\upsilon}_t, \omega) \geqslant C_t$ is:*

$$\overline{\alpha}_t := \frac{\sigma_\omega^2}{F}\left(1 + \sqrt{1 - C_t / C_t^*}\right), \tag{9}$$

where $F = \frac{2\varphi^2 R_{\max}}{(1-\gamma)^2}\left(1 + \frac{2\gamma}{\pi(1-\gamma)}\right)$ and $C_t^* = \frac{\sigma_\omega^2 \|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega)\|^2}{2F}$.

Proof. This is just a slight adaptation of existing results from [48]. Since the fixed-variance Gaussian policy is smoothing [Lemma 15 from 48], we plug its smoothing constants (7) into (8) [Theorem 9 from 48] to obtain, for all $\alpha \in \mathbb{R}$:

$$J(\boldsymbol{\upsilon}_{t+1}, \omega) - J(\boldsymbol{\upsilon}_t, \omega) \geqslant \alpha \left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega)\right\|^2 - \alpha^2 \frac{F}{2\sigma_\omega^2} \left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega)\right\|^2 := f(\alpha). \tag{20}$$

Thus, imposing $f(\alpha) \geqslant C_t$ is enough to ensure $J(\boldsymbol{\upsilon}_{t+1}, \omega) - J(\boldsymbol{\upsilon}_t, \omega) \geqslant C_t$. This yields a second-order inequality in α, whose solution is:

$$\frac{\sigma_\omega^2}{F}\left(1 - \sqrt{1 - \frac{C_t}{C_t^*}}\right) \leqslant \alpha \leqslant \frac{\sigma_\omega^2}{F}\left(1 + \sqrt{1 - \frac{C_t}{C_t^*}}\right), \tag{21}$$

provided $\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega) \neq 0$, which is a reasonable assumption (if $\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega) = 0$, the algorithm has already converged and any update is void), and $C_t \leqslant C_t^*$, which is true by hypothesis. To conclude the proof, we just select the largest step size satisfying (21). □

Lemma 4.1. *Let Π_Ω be the class of Gaussian policies parametrized as in* (4)*, but with* fixed mean parameter *$\boldsymbol{\upsilon}$. Π_Ω is $\left(\frac{4}{\sqrt{2\pi e}}, 2, 2\right)$-smoothing.*

Proof. The definition of smoothing policies is from [48, Definition 4] and reported in (6). Since the mean parameter $\boldsymbol{\upsilon}$ is fixed, the policy parameter space can be restricted to Ω. Recall that $\sigma_\omega = e^\omega$. We need the following derivatives:

$$\begin{aligned}\nabla_\omega \log \pi_{\boldsymbol{\upsilon},\omega}(a|s) &= \nabla_\omega\left(-\omega - \frac{1}{2}e^{-2\omega}(a - \mu_{\boldsymbol{\upsilon}}(s))^2\right) \\ &= e^{-2\omega}(a - \mu_{\boldsymbol{\upsilon}}(s))^2 - 1, &(22)\\ \nabla_\omega \nabla_\omega^T \log \pi_{\boldsymbol{\upsilon},\omega}(a|s) &= -2e^{-2\omega}(a - \mu_{\boldsymbol{\upsilon}}(s))^2. &(23)\end{aligned}$$

Let $x := e^{-\omega}(a - \mu_{\boldsymbol{v}}(s))$ in the following, and note that $\nabla_\omega x = e^{-\omega}$. First we compute ψ:

$$\begin{aligned}
&\sup_{s\in\mathcal{S}} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{v},\omega}} \left[\|\nabla_\omega \log \pi_{\boldsymbol{v},\omega}(a|s)\|\right] \\
&\qquad = \sup_{s\in\mathcal{S}} \frac{e^{-\omega}}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-\frac{1}{2}e^{-2\omega}(a-\mu_{\boldsymbol{v}}(s))^2} \left|e^{-2\omega}(a-\mu_{\boldsymbol{v}}(s))^2 - 1\right| \mathrm{d}a \\
&\qquad = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-x^2/2} |x^2 - 1| \mathrm{d}x \\
&\qquad = \frac{4}{\sqrt{2\pi e}} := \psi.
\end{aligned} \tag{24}$$

Next, we compute κ:

$$\begin{aligned}
&\sup_{s\in\mathcal{S}} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{v},\omega}} \left[\|\nabla_\omega \log \pi_{\boldsymbol{v},\omega}(a|s)\|^2\right] \\
&\qquad = \sup_{s\in\mathcal{S}} \frac{e^{-\omega}}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-\frac{1}{2}e^{-2\omega}(a-\mu_{\boldsymbol{v}}(s))^2} \left(e^{-2\omega}(a-\mu_{\boldsymbol{v}}(s))^2 - 1\right)^2 \mathrm{d}a \\
&\qquad = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-x^2/2} (x^2 - 1)^2 \mathrm{d}x = 2 := \kappa.
\end{aligned} \tag{25}$$

Finally, we compute ξ:

$$\begin{aligned}
&\sup_{s\in\mathcal{S}} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{v},\omega}} \left[\left\|\nabla_\omega \nabla_\omega^T \log \pi_{\boldsymbol{v},\omega}(a|s)\right\|\right] \\
&\qquad = \sup_{s\in\mathcal{S}} \frac{e^{-\omega}}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-\frac{1}{2}e^{-2\omega}(a-\mu_{\boldsymbol{v}}(s))^2} \left|-2e^{-2\omega}(a-\mu_{\boldsymbol{v}}(s))^2\right| \mathrm{d}a \\
&\qquad = \frac{2}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-x^2/2} x^2 \mathrm{d}x = 2 := \xi.
\end{aligned} \tag{26}$$

Indeed, the computed constants are independent from the value of ω. □

Theorem 4.2. *Let Π_Ω be the class of policies defined in Lemma 4.1. Let $\omega_t \in \Omega$ and $\omega_{t+1} \leftarrow \omega_t + \beta_t \nabla_\omega J(\boldsymbol{v}, \omega_t)$. For any $C_t \leqslant C_t^*$, the largest step-size satisfying* (5) *is:*

$$\overline{\beta}_t = \frac{1}{G}\left(1 + \sqrt{1 - C_t/C_t^*}\right), \tag{14}$$

where $C_t^* = \frac{\|\nabla_\omega J(\boldsymbol{v},\omega_t)\|^2}{2G}$ and $G = \frac{4R_{\max}}{(1-\gamma)^2}\left(1 + \frac{4\gamma}{\pi e(1-\gamma)}\right)$.

Proof. Similarly to the proof of Lemma 2.1, we plug the smoothing constants from Lemma 4.1 into (8) to obtain:

$$J(\boldsymbol{v}, \omega_{t+1}) - J(\boldsymbol{v}, \omega_t) \geqslant \beta \|\nabla_\omega J(\boldsymbol{v}, \omega_t)\|^2 - \beta^2 \frac{G}{2} \|\nabla_\omega J(\boldsymbol{v}_t, \omega)\|^2 := f(\beta). \tag{27}$$

Solving $f(\beta) \geqslant C_t$ yields:

$$\frac{1}{G}\left(1 - \sqrt{1 - \frac{C_t}{C_t^*}}\right) \leqslant \beta \leqslant \frac{1}{G}\left(1 + \sqrt{1 - \frac{C_t}{C_t^*}}\right), \tag{28}$$

where, again, we assume the policy gradient is non-zero. The proof is concluded by selecting the largest step size satisfying (28). □

Theorem 4.3. *Let Π_Θ be a (ψ, κ, ξ)-smoothing policy class, $\boldsymbol{\theta}_t \in \Theta$, and $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta_t \boldsymbol{x}_t$, where $\boldsymbol{x}_t \in \mathbb{R}^m$ and $\eta_t \in \mathbb{R}$ is a (possibly negative) step size. Let $\lambda_t := \frac{\langle \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t), \boldsymbol{x}_t \rangle}{\|\boldsymbol{x}_t\|}$ be the scalar projection of $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t)$ onto $\boldsymbol{x}_t$. For any $C_t \leqslant C_t^*$, provided $\lambda_t \neq 0$, the largest step size guaranteeing $J(\boldsymbol{\theta}_{t+1}) - J(\boldsymbol{\theta}_t) \geqslant C_t$ is:*

$$\overline{\eta}_t = \frac{|\lambda_t|}{L\|\boldsymbol{x}_t\|}\left(sign(\lambda_t) + \sqrt{1 - C_t/C_t^*}\right), \tag{15}$$

where $C_t^* = \frac{\lambda_t^2}{2L}$ and $L = \frac{R_{\max}}{(1-\gamma)^2}\left(\frac{2\gamma\psi^2}{1-\gamma} + \kappa + \xi\right)$.

Proof. Since we are no longer dealing with a policy gradient update, we need a generalization of (8). From [48, Theorem 8]:

$$J(\boldsymbol{\theta}_t + \Delta\boldsymbol{\theta}) - J(\boldsymbol{\theta}_t) \geqslant \langle \Delta\boldsymbol{\theta}, \nabla J(\boldsymbol{\theta}_t)\rangle - \frac{L}{2}\left\|\Delta\boldsymbol{\theta}\right\|^2, \tag{29}$$

for any parameter update $\Delta\boldsymbol{\theta} \in \mathbb{R}^m$. In our case:

$$J(\boldsymbol{\theta}_{t+1}) - J(\boldsymbol{\theta}_t) \geqslant \eta_t \langle \boldsymbol{x}_t, \nabla J(\boldsymbol{\theta}_t)\rangle - \eta_t^2 \frac{L}{2}\left\|\boldsymbol{x}_t\right\|^2 := f(\eta_t), \tag{30}$$

where $\langle\cdot,\cdot\rangle$ denotes the dot product. Intuitively, the more $\boldsymbol{x}_t$ agrees with the improvement direction $\nabla J(\boldsymbol{\theta}_t)$, the more improvement can be guaranteed. We first assume $\langle \boldsymbol{x}_t, \nabla J(\boldsymbol{\theta}_t)\rangle \neq 0$. Solving $f(\eta_t) \geqslant C_t$ yields:

$$\frac{|\lambda_t|}{L\left\|\boldsymbol{x}_t\right\|}\left(sign(\lambda_t) - \sqrt{1 - \frac{C_t}{C_t^*}}\right) \leqslant \eta_t \leqslant \frac{|\lambda_t|}{L\left\|\boldsymbol{x}_t\right\|}\left(sign(\lambda_t) + \sqrt{1 - \frac{C_t}{C_t^*}}\right), \tag{31}$$

from which we select the largest safe step size. For $C_t \geqslant 0$, depending on the sign of λ_t (i.e., whether $\boldsymbol{x}_t$ agrees with the gradient or not) and on the value of C_t, the step size may be non-positive. Intuitively, if a positive improvement is required but $\boldsymbol{x}_t$ is pejorative, a negative step size is used to invert it. Instead, if $C_t < 0$ (bounded worsening), a small-enough step in the direction of $\boldsymbol{x}_t$ is always acceptable.

We now consider the special case $\langle \boldsymbol{x}_t, \nabla J(\boldsymbol{\theta}_t)\rangle = 0$, i.e., $\lambda = 0$. In this case, only non-positive values of C_t are allowed. Intuitively, $\boldsymbol{x}_t$ is orthogonal to the improvement direction, so no positive improvement can be guaranteed. Under the restriction $C_t \leqslant 0$, the following range of step sizes is safe:

$$-\frac{1}{\left\|\boldsymbol{x}_t\right\|}\sqrt{-\frac{2C_t}{L}} \leqslant \eta_t \leqslant \frac{1}{\left\|\boldsymbol{x}_t\right\|}\sqrt{-\frac{2C_t}{L}}, \tag{32}$$

from which we select $\overline{\eta}_t = \frac{1}{\|\boldsymbol{x}_t\|}\sqrt{-\frac{2C_t}{L}}$. □

B Meta Gradient Estimation

In this section, we propose an unbiased estimator for $\nabla_\omega \left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\|$, which is necessary to estimate $\nabla_\omega \mathcal{L}(\boldsymbol{\theta})$ (recall that $\boldsymbol{\theta} = [\boldsymbol{\upsilon}, \omega]$ is the full vector of policy parameters). First note that:

$$\nabla_\omega \left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\| = \frac{\langle \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}), \nabla_\omega \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\rangle}{\left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\|}, \tag{33}$$

which is the scalar projection of $\nabla_\omega \nabla_{\boldsymbol{\upsilon}} J$ onto $\nabla_{\boldsymbol{\upsilon}} J$.

An estimator for $\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})$ is already available (2). We now show how to estimate $\nabla_\omega \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. First note that:

$$\begin{aligned}\nabla_\omega \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau) &= \sum_{h=0}^{H} \nabla_\omega \nabla_{\boldsymbol{\upsilon}} \log \pi_{\boldsymbol{\theta}}(a_h \mid s_h) = \sum_{h=0}^{H} \nabla_\omega \frac{a_h - \mu_{\boldsymbol{\theta}}(s_h)}{e^{2\omega}} = -2\sum_{t=0}^{H} \frac{a - \mu_{\boldsymbol{\theta}}}{e^{2\omega}} \\ &= -2\nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau). \end{aligned} \tag{34}$$

Using the log-trick:

$$\begin{aligned}\nabla_\omega \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}) &= \nabla_\omega \mathop{\mathbb{E}}_{\tau\sim p_{\boldsymbol{\theta}}}\left[\nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau) R(\tau)\right] \\ &= \mathop{\mathbb{E}}_{\tau\sim p_{\boldsymbol{\theta}}}\left[\nabla_\omega \log p_{\boldsymbol{\theta}}(\tau) \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau) R(\tau)\right] + \mathop{\mathbb{E}}_{\tau\sim p_{\boldsymbol{\theta}}}\left[\nabla_\omega \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau) R(\tau)\right] \\ &= \mathop{\mathbb{E}}_{\tau\sim p_{\boldsymbol{\theta}}}\left[\nabla_\omega \log p_{\boldsymbol{\theta}}(\tau) \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau) R(\tau)\right] - 2\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}) \\ &:= mix(\boldsymbol{\theta}) - 2\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}). \end{aligned} \tag{35}$$

Since we can reuse the policy gradient w.r.t. $\boldsymbol{\upsilon}$ in (35), we have reduced the problem of estimating $\nabla_\omega \mathcal{L}(\boldsymbol{\theta})$ to that of estimating $mix(\boldsymbol{\theta}) := \mathbb{E}_{\tau\sim p_{\boldsymbol{\theta}}}\left[\nabla_\omega \log p_{\boldsymbol{\theta}}(\tau) \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau) R(\tau)\right]$. The following estimator is inspired by GPOMDP [7]:

Theorem B.1. *An unbiased estimator for $mix(\boldsymbol{\theta}) := \mathbb{E}_{\tau\sim p_{\boldsymbol{\theta}}}\left[\nabla_\omega \log p_{\boldsymbol{\theta}}(\tau)\nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau)R(\tau)\right]$ is:*

$$\widehat{mix}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{h=0}^{H}\gamma^h r_h^n \left(\sum_{i=0}^{h}\nabla_\omega \log \pi_{\boldsymbol{\theta}}(a_i^n|s_i^n)\right)\left(\sum_{j=0}^{h}\nabla_{\boldsymbol{\upsilon}} \log \pi_{\boldsymbol{\theta}}(a_j^n|s_j^n)\right), \tag{36}$$

where subscripts denote time steps and superscripts denote trajectories. To preserve the unbiasedness of the estimator, separate trajectories must be used to compute the two inner sums.

Proof. Let's abbreviate action probabilities as $\pi_k = \pi_{\boldsymbol{\theta}}(a_k|s_k)$ and sub-trajectory probabilities as $p_{\boldsymbol{\theta}}(\tau_{h:k}) = \pi_{\boldsymbol{\theta}}(a_h|s_h)\mathcal{P}(s_{h+1}|s_h,a_h)\dots\mathcal{P}(s_k|s_{k-1},a_{k-1})$. We can split $mix(\boldsymbol{\theta})$ into the sum of four components:

$$\begin{aligned}
mix\,(\boldsymbol{\theta}) &= \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\nabla_\omega \log p_{\boldsymbol{\theta}}(\tau)\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\tau)R(\tau)\mathrm{d}\tau = \\
&= \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\left(\sum_{h=0}^{H}\gamma^h r_h\right)\left(\sum_{i=0}^{H}\nabla_\omega \log \pi_i\right)\left(\sum_{j=0}^{H}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau = \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\left(\sum_{j=0}^{h}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau && (37)\\
&+ \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\left(\sum_{j=h+1}^{H}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau && (38)\\
&+ \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\gamma^h r_h\left(\sum_{i=h+1}^{H}\nabla_\omega \log \pi_i\right)\left(\sum_{j=0}^{h}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau && (39)\\
&+ \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\gamma^h r_h\left(\sum_{i=h+1}^{H}\nabla_\omega \log \pi_i\right)\left(\sum_{j=h+1}^{H}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau. && (40)
\end{aligned}$$

Next, we show that (38), (39) and (40) all evaluate to 0:

$$\begin{aligned}
(38) &= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\left(\sum_{j=h+1}^{H}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\mathrm{d}\tau\int_T p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\left(\sum_{j=h+1}^{H}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\mathrm{d}\tau\int_T p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\mathrm{d}\tau \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\mathrm{d}\tau\int_T \nabla_{\boldsymbol{\upsilon}} p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\mathrm{d}\tau \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right)\gamma^h r_h\left(\sum_{i=0}^{h}\nabla_\omega \log \pi_i\right)\mathrm{d}\tau\nabla_{\boldsymbol{\upsilon}}\int_T p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\mathrm{d}\tau \\
&= 0.
\end{aligned}$$

Analogously, we can say that (39) = 0. Finally:

$$\begin{aligned}
(40) &= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:H}\right)\gamma^h r_h\left(\sum_{i=h+1}^{H}\nabla_\omega \log \pi_i\right)\left(\sum_{j=h+1}^{H}\nabla_{\boldsymbol{\upsilon}} \log \pi_j\right)\mathrm{d}\tau \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right)\gamma^h r_h\mathrm{d}\tau\int_T p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\nabla_\omega \log p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}\left(\tau_{h+1:H}\right)\mathrm{d}\tau \\
&= \sum_{h=0}^{H}\int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right)\gamma^h r_h\mathrm{d}\tau\int_T\left(\nabla_\omega\nabla_{\boldsymbol{\upsilon}} p_{\boldsymbol{\theta}}(\tau_{h+1:H}) - \nabla_\omega\nabla_{\boldsymbol{\upsilon}}\log p_{\boldsymbol{\theta}}(\tau_{h+1:H})\right)\mathrm{d}\tau
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{h=0}^{H} \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right) \gamma^h r_h \mathrm{d}\tau \int_T \nabla_\omega \nabla_{\boldsymbol{\upsilon}} p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \mathrm{d}\tau \\
&\qquad - \sum_{h=0}^{H} \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right) \gamma^h r_h \mathrm{d}\tau \int_T \nabla_\omega \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \mathrm{d}\tau \\
&= \sum_{h=0}^{H} \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right) \gamma^h r_h d\tau \nabla_\omega \nabla_{\boldsymbol{\upsilon}} \int_T p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \mathrm{d}\tau \\
&\qquad - \sum_{h=0}^{H} \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right) \gamma^h r_h \mathrm{d}\tau \int_T -2 p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \mathrm{d}\tau \\
&= 2 \sum_{h=0}^{H} \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right) \gamma^h r_h \mathrm{d}\tau \int_T p_{\boldsymbol{\theta}}(\tau) \nabla_{\boldsymbol{\upsilon}} \log p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \mathrm{d}\tau \\
&= 2 \sum_{h=0}^{H} \int_T p_{\boldsymbol{\theta}}\left(\tau_{0:h}\right) \gamma^h r_h \mathrm{d}\tau \int_T \nabla_{\boldsymbol{\upsilon}} p_{\boldsymbol{\theta}}(\tau_{h+1:H}) \mathrm{d}\tau = 0.
\end{aligned}
$$

Hence, $mix(\boldsymbol{\theta})$ is equal to (37) alone, of which the proposed $\widehat{mix}(\boldsymbol{\theta})$ is a Monte Carlo estimator.

□

Similarly to what has been done for GPOMDP [49], we can introduce a baseline to reduce the variance of the estimator. Let:

$$
\widehat{mix}_h(\boldsymbol{\theta}) = \mathbb{E}\left[\underbrace{\left(\sum_{k=0}^{h} \nabla_{\boldsymbol{\upsilon}} \log \pi_k\right)}_{G_h} \underbrace{\left(\sum_{k=0}^{h} \nabla_\omega \log \pi_k\right)}_{H_h} \left(\underbrace{\gamma^h r_h}_{F_h} - b_h \right) \right], \tag{41}
$$

where b_h is a generic baseline that is independent from actions a_k. Any baseline $b_h = \widetilde{b_h}\left(\frac{G_h + H_h}{G_h H_h}\right)$ will keep the estimator unbiased for any value of $\widetilde{b_h}$, as long as different data are used for each multiplicative term:

$$
\begin{aligned}
E\left[G_h H_h \left(F_h - b_h\right)\right] &= E\left[G_h H_h F_h\right] - E\left[G_h H_h b_h\right] \\
&= E\left[G_h H_h F_h\right] - E\left[G_h H_h \widetilde{b_h} \left(\frac{G_h + H_h}{G_h H_h}\right)\right] \\
&= E\left[G_h H_h F_h\right] - \widetilde{b_h} E\left[G_h\right] - \widetilde{b_h}\left[H_h\right] \\
&= E\left[G_h H_h F_h\right].
\end{aligned}
$$

We choose $\widetilde{b_h}$ as to minimize the variance of $\widehat{mix}_h$:

$$
\begin{aligned}
Var[\widehat{mix}_h] &= E\left[\widehat{mix}_h^{\,2}\right] - E\left[\widehat{mix}_h\right]^2 \\
&= E\left[G_h^2 H_h^2 \left(F_h^2 - 2F_h \widetilde{b_h} \frac{G_h + H_h}{G_h H_h} + \widetilde{b_h}^{\,2} \frac{(G_h + H_h)^2}{G_h^2 H_h^2}\right)\right] - E\left[G_h H_h F_h\right]^2 \\
&= E\left[G_h^2 H_h^2 F_h^2\right] - 2\widetilde{b_h} E\left[G_h H_h F_h \left(G_h + H_h\right)\right] + \widetilde{b_h}^{\,2} E\left[\left(G_h + H_h\right)^2\right] \\
&\quad - E\left[G_h H_h F_h\right]^2.
\end{aligned}
$$

Setting the gradient to zero yields:

$$
b_h^* = \arg\min_{\widetilde{b_h}} Var\left[\widehat{mix}_h\right] = \frac{E\left[G_h H_h F_h \left(G_h + H_h\right)\right]}{E\left[\left(G_h + H_h\right)^2\right]}.
$$

Hence the estimator has minimum variance with baseline:

$$
b_h = b_h^* \frac{G_h + H_h}{G_h H_h} = \frac{G_h H_h F_h \left(G_h + H_h\right)}{\left(G_h + H_h\right)^2} \frac{G_h + H_h}{G_h H_h},
$$

which can be estimated from samples as in [49].

Finally:

$$\begin{aligned}\widehat{\nabla}_\omega \left\|\nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\| &= \frac{\left\langle \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}), \widehat{\nabla}_\omega \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}) \right\rangle}{\left\|\widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\|}, \\ &= \frac{\left\langle \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}), \widehat{mix}(\boldsymbol{\theta}) - 2\widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}) \right\rangle}{\left\|\widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\|} \\ &= \frac{\left\langle \widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta}), \widehat{mix}(\boldsymbol{\theta}) \right\rangle}{\left\|\widehat{\nabla}_{\boldsymbol{\upsilon}} J(\boldsymbol{\theta})\right\|} - 2. \end{aligned} \tag{42}$$

To preserve the unbiasedness of this estimator we need to employ three independent sets of sample trajectories: two for $\widehat{mix}$ and a third one for the (normalized) policy gradient estimator w.r.t. $\boldsymbol{\upsilon}$. For the latter, we can re-use the same data used to compute the other additive terms of $\widehat{\nabla}_\omega \mathcal{L}$. Additional, independent data are needed for the variance-minimizing baseline. However, as often in practice, the bias introduced by using a single batch of trajectories to compute the estimator (and its baseline) is too small to justify the variance introduced by splitting the batch in order to preserve unbiasedness. Hence, we never split our batches in the experiments.

C Approximate Framework Revisited

In this section, we restate the results of Section 5 in a more formal way and prove them.

In the following, let $\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J$, $\widehat{\nabla}^N_\omega J$ and $\widehat{\nabla}^N_\omega \mathcal{L}$ be unbiased estimators of $\nabla_{\boldsymbol{\upsilon}} J$, $\nabla_{\boldsymbol{\upsilon}} J$ and $\nabla_\omega \mathcal{L}$, respectively, each using a batch of N trajectories.

Corollary C.1. *Let Π_Υ be the class of Gaussian policies parametrized as in* (4)*, but with* fixed variance parameter *ω. Let $\boldsymbol{\upsilon}_t \in \mathbb{R}^m$ and $\boldsymbol{\upsilon}_{t+1} = \boldsymbol{\upsilon}_t + \alpha_t \widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega)$. Under Assumption 5.1, provided $N > \epsilon_\delta^2 / \left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)\right\|^2$, for any $C_t \leqslant C_t^*$, the largest step size guaranteeing $J(\boldsymbol{\upsilon}_{t+1}, \omega) - J(\boldsymbol{\upsilon}_t, \omega) \geqslant C_t$ is:*

$$\widetilde{\alpha}_t = \frac{\sigma_\omega^2 \left(\left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)\right\| - \frac{\epsilon_\delta}{\sqrt{N}} \right)}{F \left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)\right\|} \left(1 + \sqrt{1 - \frac{C_t}{C_t^*}} \right), \tag{43}$$

where $C_t^* = \frac{\sigma_{\omega_t}^2 \left(\left\|\widehat{\nabla}^N_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega_t)\right\| - \frac{\epsilon_\delta}{\sqrt{N}} \right)^2}{2F}$ and F is from Lemma 2.1.

Proof. From (29) with $\Theta = \Upsilon$ and $\Delta\boldsymbol{\theta} = \alpha_t \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega)$ we have:

$$\begin{aligned}
J(\boldsymbol{\upsilon}_{t+1}, \omega) - J(\boldsymbol{\upsilon}_t, \omega) &\geqslant \alpha_t \left\langle \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega), \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega) \right\rangle - \alpha_t^2 \frac{L}{2} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 \\
&= \alpha_t \left\langle \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega), \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega) \pm \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\rangle \\
&\quad - \alpha_t^2 \frac{L}{2} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 \\
&= \alpha_t \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 + \alpha_t \left\langle \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega), \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega) - \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\rangle \\
&\quad - \alpha_t^2 \frac{L}{2} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 \\
&\geqslant \alpha_t \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 - \alpha_t \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\| \left\| \nabla_{\boldsymbol{\upsilon}} J(\boldsymbol{\upsilon}_t, \omega) - \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\| \\
&\quad - \alpha_t^2 \frac{L}{2} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 \qquad (44) \\
&\geqslant \alpha_t \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 - \alpha_t \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\| \frac{\epsilon_\delta}{\sqrt{N}} \\
&\quad - \alpha_t^2 \frac{L}{2} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2 \qquad (45) \\
&\geqslant \alpha_t \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\| \left(\left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\| - \frac{\epsilon_\delta}{\sqrt{N}} \right) \\
&\quad - \alpha_t^2 \frac{L}{2} \left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\|^2, \qquad (46)
\end{aligned}$$

where (44) is from Cauchy-Schwartz inequality and (45) is from Assumption 5.1. The hypothesis on the batch size makes the $\left(\left\| \widehat{\nabla}_{\boldsymbol{\upsilon}}^N J(\boldsymbol{\upsilon}_t, \omega) \right\| - \frac{\epsilon_\delta}{\sqrt{N}} \right)$ term positive. We then proceed as in the proof of Lemma 2.1. □

Corollary C.2. *Let Π_Ω be the class of policies defined in Lemma 4.1. Let $\omega_t \in \Omega$ and $\omega_{t+1} \leftarrow \omega_t + \eta_t \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{\upsilon}, \omega_t)$. Under Assumption 5.1, provided $\widehat{\lambda}_t \neq 0$ and $N > \epsilon_\delta^2 / \widehat{\lambda}_t^2$, for any $C_t \leqslant C_t^*$, the largest step-size satisfying* (5) *is:*

$$\widetilde{\eta}_t = \frac{\left| \widehat{\lambda}_t \right| - \frac{\epsilon_\delta}{\sqrt{N}}}{G \left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) \right\|} \left(sign\left(\widehat{\lambda}_t \right) + \sqrt{1 - \frac{C_t}{C_t^*}} \right), \qquad (47)$$

where $\widehat{\lambda}_t := \frac{\left\langle \widehat{\nabla}_\omega J(\boldsymbol{\upsilon}_t, \omega_t), \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) \right\rangle}{\left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t) \right\|}$ is the scalar projection of $\widehat{\nabla}_\omega J(\boldsymbol{\upsilon}_t, \omega_t)$ onto $\widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{\upsilon}_t, \omega_t)$, $C_t^* = \frac{\left(|\widehat{\lambda}_t| - \frac{\epsilon_\delta}{\sqrt{N}} \right)^2}{2G}$, and G is from Theorem 4.2.

Proof. From (29) with $\Theta = \Omega$ and $\Delta\boldsymbol{\theta} = \eta_t \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t)$ we have:

$$
\begin{aligned}
J(\boldsymbol{v}, \omega_{t+1}) - J(\boldsymbol{v}, \omega_t) &\geqslant \eta_t \left\langle \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t), \nabla_\omega J(\boldsymbol{v}, \omega_t) \right\rangle - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2 \\
&= \eta_t \left\langle \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t), \nabla_\omega J(\boldsymbol{v}, \omega_t) \pm \widehat{\nabla}_\omega^N J(\boldsymbol{v}, \omega_t) \right\rangle - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2 \\
&= \eta_t \left\langle \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t), \widehat{\nabla}_\omega^N J(\boldsymbol{v}, \omega_t) \right\rangle \\
&\quad + \eta_t \left\langle \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t), \nabla_\omega J(\boldsymbol{v}, \omega_t) - \widehat{\nabla}_\omega^N J(\boldsymbol{v}, \omega_t) \right\rangle \\
&\quad - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2 \\
&\geqslant \eta_t \left\langle \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t), \widehat{\nabla}_\omega^N J(\boldsymbol{v}, \omega_t) \right\rangle \\
&\quad - |\eta_t| \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\| \left\| \nabla_\omega J(\boldsymbol{v}, \omega_t) - \widehat{\nabla}_\omega^N J(\boldsymbol{v}, \omega_t) \right\| \\
&\quad - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2 &(48) \\
&\geqslant \eta_t \left\langle \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t), \widehat{\nabla}_\omega^N J(\boldsymbol{v}, \omega_t) \right\rangle - |\eta_t| \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\| \frac{\epsilon_\delta}{\sqrt{N}} \\
&\quad - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2 &(49) \\
&\geqslant \eta_t \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\| \left(\widehat{\lambda}_t - sign(\eta_t) \frac{\epsilon_\delta}{\sqrt{N}} \right) \\
&\quad - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2, &(50)
\end{aligned}
$$

where (48) is from Cauchy-Schwartz inequality and (49) is from Assumption 5.1. Note the absolute value on η_t, which may be negative. We first consider the case $\eta_t > 0$, which yields:

$$
\begin{aligned}
J(\boldsymbol{v}, \omega_{t+1}) - J(\boldsymbol{v}, \omega_t) &\geqslant \eta_t \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\| \left(\widehat{\lambda}_t - \frac{\epsilon_\delta}{\sqrt{N}} \right) \\
&\quad - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2. \qquad (51)
\end{aligned}
$$

Solving the safety constraint for $\widetilde{\eta}_t$ yields:

$$
\widetilde{\eta}_t = \frac{1}{G \left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t) \right\|} \left(\widehat{\lambda}_t - \frac{\epsilon_\delta}{\sqrt{N}} + \left| \widehat{\lambda}_t - \frac{\epsilon_\delta}{\sqrt{N}} \right| \sqrt{1 - \frac{C_t}{C_t^*}} \right). \qquad (52)
$$

Given the batch size condition, the step size is indeed positive if and only if $\widehat{\lambda}_t > 0$. We then consider the case $\eta_t \leqslant 0$, which yields:

$$
\begin{aligned}
J(\boldsymbol{v}, \omega_{t+1}) - J(\boldsymbol{v}, \omega_t) &\geqslant \eta_t \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\| \left(\widehat{\lambda}_t + \frac{\epsilon_\delta}{\sqrt{N}} \right) \\
&\quad - \eta_t^2 \frac{L}{2} \left\| \widehat{\nabla}_\omega^N \mathcal{L}(\boldsymbol{v}, \omega_t) \right\|^2. \qquad (53)
\end{aligned}
$$

Solving the safety constraint for $\widetilde{\eta}_t$ yields:

$$
\widetilde{\eta}_t = \frac{1}{G \left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t) \right\|} \left(\widehat{\lambda}_t + \frac{\epsilon_\delta}{\sqrt{N}} + \left| \widehat{\lambda}_t + \frac{\epsilon_\delta}{\sqrt{N}} \right| \sqrt{1 - \frac{C_t}{C_t^*}} \right). \qquad (54)
$$

Given the batch size condition, the step size is indeed non-positive if and only if $\widehat{\lambda}_t < 0$. The two cases can be unified as:

$$
\overline{\eta}_t = \begin{cases} \frac{\widehat{\lambda}_t - \frac{\epsilon_\delta}{\sqrt{N}}}{G \left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t) \right\|} \left(1 + \sqrt{1 - \frac{C_t}{C_t^*}} \right) & \text{if } \widehat{\lambda}_t > 0, \\ \frac{\widehat{\lambda}_t + \frac{\epsilon_\delta}{\sqrt{N}}}{G \left\| \widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t) \right\|} \left(1 - \sqrt{1 - \frac{C_t}{C_t^*}} \right) & \text{if } \widehat{\lambda}_t < 0, \end{cases} \qquad (55)
$$

which can be further simplified to obtain (47).

As in the exact framework, we can treat the case $\widehat{\lambda}_t = 0$ separately. Under the restriction $C_t \leqslant 0$, the following range of step sizes is safe:

$$-\frac{1}{\left\|\widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t)\right\|}\sqrt{-\frac{2C_t}{G}} \leqslant \eta_t \leqslant \frac{1}{\left\|\widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t)\right\|}\sqrt{-\frac{2C_t}{G}}, \tag{56}$$

from which we select $\overline{\eta}_t = \frac{1}{\left\|\widehat{\nabla}_\omega \mathcal{L}(\boldsymbol{v}_t, \omega_t)\right\|}\sqrt{-\frac{2C_t}{G}}$. No assumption on the batch size is requested, but only non-positive improvement constraints can be satisfied. □

D Characterizing the Estimation Error

The safe step sizes for the approximate framework presented in Section 5 require an upper bound on the policy gradient estimator error. For simplicity and generality, in this section we will not distinguish between mean and variance parameters. Thus, we seek an $\epsilon_\delta > 0$ such that, for all $\delta \in (0, 1)$ and $N \geqslant 1$:

$$\left\|\widehat{\nabla}^N J(\boldsymbol{\theta}_t) - \nabla J(\boldsymbol{\theta}_t)\right\| \leqslant \frac{\epsilon_\delta}{\sqrt{N}}, \tag{57}$$

with probability at least $1 - \delta$, where $\widehat{\nabla}^N J(\boldsymbol{\theta}_t)$ is an *unbiased* estimator of $\nabla J(\boldsymbol{\theta}_t)$ employing N sample trajectories. A formal way to obtain such an ϵ_δ, based on Chebychev's inequality and an upper bound on the variance of GPOMDP [71] is provided in [48]. However, this solution tends to be over-conservative [47]. With a small additional assumption, i.e., Gaussianity of $\widehat{\nabla} J(\boldsymbol{\theta}_t)$, we can use Gaussian confidence regions instead[9]. Since we care about the magnitude error of an m-dimensional random vector, we propose to employ ellipsoidal confidence regions. For any $\delta \in (0, 1)$, let E_δ be the following set:

$$E_\delta = \left\{\boldsymbol{x} \in \mathbb{R}^m : \left(\widehat{\nabla}^N J(\boldsymbol{\theta}_t) - \boldsymbol{x}\right)^T S^{-1} \left(\widehat{\nabla}^N J(\boldsymbol{\theta}_t) - \boldsymbol{x}\right) < \frac{m}{N-m} F_{1-\delta,m,N-m}\right\}, \tag{58}$$

where S is the sample covariance of $\widehat{\nabla}^1 J(\boldsymbol{\theta}_t)$ and $F_{1-\delta,m,N-m}$ is the quantile $(1 - \delta)$ of the F-distribution with m and $n - m$ degrees of freedom. This set is centered in $\widehat{\nabla}^N J(\boldsymbol{\theta}_t)$ and is delimited by an ellipsoid. It is a standard result [25] that, with probability $1 - \delta$, the true gradient is contained in this region, i.e., $\mathbb{P}\left(\nabla J(\boldsymbol{\theta}_t) \in E_\delta\right) = 1 - \delta$. Equivalently, the difference $\widehat{\nabla}^N J(\boldsymbol{\theta}_t) - \nabla J(\boldsymbol{\theta}_t)$ is contained within the following origin-centered ellipsoid:

$$\mathcal{E}_\delta = \left\{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{x}^T A_\delta \boldsymbol{x} = 1\right\}, \tag{59}$$

where $A_\delta = \left(\frac{mF_{1-\delta,d,N-m}}{N-m} S\right)^{-1}$. Thus, the estimation error $\left\|\widehat{\nabla}^N J(\boldsymbol{\theta}_t) - \nabla J(\boldsymbol{\theta}_t)\right\|$ cannot be larger than the largest semi-axis of $\mathcal{E}_\delta$. Simple algebraic computations yield the following:

$$\left\|\widehat{\nabla}^N J(\boldsymbol{\theta}_t) - \nabla J(\boldsymbol{\theta}_t)\right\| \leqslant \sqrt{\frac{mF_{1-\delta,m,n-m}\left\|S\right\|}{N-m}}, \tag{60}$$

with probability at least $1 - \delta$, where $\|S\|$ denotes the spectral norm (i.e., the largest eigenvalue) of the sample covariance. The latter can be computed efficiently with the Lanczos method [34]. Finally, we define the following error bound:

$$\epsilon_\delta = \sqrt{\frac{NmF_{1-\delta,m,n-m}\left\|S\right\|}{N-m}}, \tag{61}$$

which can be directly used in Algorithm 2.

E Extensions

In this section, we consider possible generalizations of the results of the paper.

[9]The applicability of this assumption relies on the Central Limit Theorem, hence is only justified by sufficiently large batch sizes.

E.1 Multi-dimensional actions

In the main paper, we only considered scalar actions, i.e., $\mathcal{A} \subseteq \mathbb{R}$. However, many continuous RL tasks involve multi-dimensional actions [10]. The natural generalization of (3) for the case $\mathcal{A} \in \mathbb{R}^d$ is a multi-variate Gaussian policy:

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{1}{(2\pi)^{d/2}|\Sigma_{\boldsymbol{\omega}}|^{1/2}} \exp\left\{-\frac{1}{2}(a - \mu_{\boldsymbol{v}}(s))^T \Sigma_{\boldsymbol{\omega}}^{-1}(a - \mu_{\boldsymbol{v}}(s))\right\}, \tag{62}$$

where $\Sigma_{\boldsymbol{\omega}}$ is a $d \times d$ covariance matrix parametrized by $\boldsymbol{\omega}$. In this case the policy parameters are $\boldsymbol{\theta} = [\boldsymbol{v}^T|\boldsymbol{\omega}^T]^T$. We denote with m_1 the dimensionality of $\boldsymbol{v}$, with m_2 the dimensionality of $\boldsymbol{\omega}$ and with $m = m_1 + m_2$ the dimensionality of the full parameter vector $\boldsymbol{\theta}$.

The simplest case is $\Sigma_\omega = e^{2\omega}\mathbb{I}$, where $\mathbb{I}$ denotes the identity matrix. This corresponds to a factored Gaussian policy where the action dimensions are independent and the same variance is used for them all. The results of the paper extend directly to this case.

A more common parametrization [18] is $\Sigma_{\boldsymbol{\omega}} = diag(\exp^{2\boldsymbol{\omega}})$, where the covariance matrix is diagonal. This corresponds to a factored Gaussian policy where the actions dimensions are independent, but a different variance is employed for each of them. It can be useful when the actions have different scales, or when the environment is more sensitive to particular actions. Again, the results on scalar actions can be generalized quite easily by considering each action dimension separately.

Finally, $\Sigma_{\boldsymbol{\omega}}$ can be a full matrix. This allows to capture correlations among different actions. A possible parametrization is $\Sigma_{\boldsymbol{\omega}} = L_{\boldsymbol{\omega}} L_{\boldsymbol{\omega}}^T$, where $L_{\boldsymbol{\omega}}$ is a lower triangular matrix with positive diagonal entries:

$$L_{\boldsymbol{\omega}} = \begin{bmatrix} e^{\omega_{11}} & \omega_{12} & \dots & \omega_{1d} \\ \omega_{21} & e^{\omega_{22}} & \dots & \omega_{2d} \\ \vdots & \ddots & \ddots & \vdots \\ \omega_{d1} & \omega_{d2} & \dots & e^{\omega_{dd}} \end{bmatrix}. \tag{63}$$

Generalizing our results to this case is non-trivial. However, full covariance matrices are rarely used in practice.

E.2 Heteroskedastic exploration

Another generalization is to make the policy variance state-dependent. This allows to concentrate the stochasticity on those regions of the state space where there is more need of exploration. We can employ a linear parametrization as done for the mean:

$$\sigma_{\boldsymbol{\omega}} = \exp\left\{\boldsymbol{\omega}^T \boldsymbol{\phi}(s)\right\}, \tag{64}$$

where $\boldsymbol{\phi}(s)$ is a vector of bounded state features, i.e., $\sup_{s \in \mathcal{S}} \|\boldsymbol{\phi}(s)\| \leqslant \varphi$. Our results extend quite easily to this case. It is enough to adjust the smoothing constants from Lemma 4.1 as follows:

$$\psi = \frac{4\varphi}{\sqrt{2\pi e}}, \qquad \kappa = \xi = 2\varphi^2. \tag{65}$$

F Task Specifications

In this section, we provide a detailed description of the environments employed by the numerical simulations of Section 6.

F.1 Linear-Quadratic Gaussian regulator (LQG)

This is a 1D continuous control task. The state is initialized uniformly at random in $[-4, 4]$, which is also the state space. The task is deterministic otherwise. The action space is $[-4, 4]$ as well. The next state is $s_{h+1} = s_h + a_h$ (linear) and the reward is $r_h = -0.9s_h^2 - 0.9a_h^2$ (quadratic). The induced goal is to bring the state in 0 with the minimum effort. The episode is always 20 steps long. A discount factor of $\gamma = 0.9$ is used for this task.

F.2 Continuous-action Cart-Pole

This is a 2D continuous control task. The goal is to balance (keep upright) a pole situated on a cart, by applying forces to the cart in the horizontal direction. The cart has a mass of $1Kg$ and the pole has a mass of 0.1Kg and is 0.5m long. The state is four dimensional and includes the cart's position x, the cart's horizontal speed $\dot{x}$, the pole's angle w.r.t. the upright position θ and the pole's angular velocity $\dot{\theta}$. The action (force) that can be applied is $a \in [-10, 10]$. The agents receives a reward of 1 for each step. All state variables are initialized uniformly at random in $[-0.05, 0.05]$. The task is deterministic otherwise. The episode terminates when the pole falls ($|\theta| > 12$ degrees), when the cart goes too far from the initial position ($|x| > 2.4$), and anyway after 1000 time steps. The control frequency is 50Hz. A discount factor of $\gamma = 0.99$ is used for this task.

G Experimental Setting

In this section, we provide further details on the experiments.

G.1 MEPG experiment (Figure 1)

The best step size for fixed-variance GPOMDP was searched among $\alpha \in \{10, 1, 0.1, 0.01\}$; $\alpha = 0.1$ turned out to be the best choice both for $\sigma_{\omega_0} = 0.5$ and $\sigma_{\omega_0} = 5$.

The best step size for adaptive-variance GPOMDP was searched among $\alpha \in \{1, 0.1, 0.01, 0.001\}$; $\alpha = 0.01$ turned out to be the best choice both for $\sigma_{\omega_0} = 0.5$ and $\sigma_{\omega_0} = 5$.

The best step sizes for MEPG were searched among $(\alpha \times \eta) \in \{10, 1, 0.1\} \times \{1, 0.1, 0.01\}$; $(1, 0.1)$ was the best choice for $\sigma_{\omega_0} = 0.5$, while $(0.1, 0.01)$ performed better in the case $\sigma_{\omega_0} = 5$.

Five random seeds were employed for the step-size selection. Average return (i.e., averaged both across learning iterations and across different seeds) was used as a metric. Step sizes causing divergence issues were discarded. This happend, e.g., with $\alpha = 10$ in the fixed-variance GPOMDP experiment and $\alpha = 1$ in the adaptive-variance GPOMDP experiment.

The final results are averaged over 10 separate random seeds. The figure also reports 95% Student's t confidence intervals.

The batch size is $N = 500$ for all algorithms.

G.2 SEPG LQG experiment (Figure 2)

All the considered algorithms tune the step sizes automatically.

A batch size of $N = 500$ and a confidence parameter of $\delta = 0.2$ were used for all the algorithms.

The results are averaged over 10 random seeds. The figure also reports 95% Student's t confidence intervals.

G.3 SEPG Cart-Pole experiment (Figure 3)

SEPG tunes the step sizes automatically.

A batch size of $N = 500$ and a confidence parameter of $\delta = 1$ were used.

The results are averaged over 5 random seeds. The figure also reports 95% Student's t confidence intervals.

The code used for the experiments is included in the supplementary materials.