Reusing Trajectories in Policy Gradients Enables Fast Convergence

Alessandro Montenegro
Politecnico di Milano, Milan, Italy
alessandro.montenegro@polimi.it

Federico Mansutti
Politecnico di Milano, Milan, Italy
federico.mansutti@mail.polimi.it

Marco Mussi
Politecnico di Milano, Milan, Italy
marco.mussi@polimi.it

Matteo Papini
Politecnico di Milano, Milan, Italy
matteo.papini@polimi.it

Alberto Maria Metelli
Politecnico di Milano, Milan, Italy
albertomaria.metelli@polimi.it

Abstract

Policy gradient (PG) methods are a class of effective *reinforcement learning* algorithms, particularly when dealing with continuous control problems. These methods learn the parameters of parametric policies via stochastic gradient ascent, typically using *on-policy* trajectory data to estimate the policy gradient. However, such reliance on fresh data makes them sample-inefficient. Indeed, vanilla PG methods require $\mathcal{O}(\epsilon^{-2})$ trajectories to reach an ϵ-approximate stationary point. A common strategy to improve efficiency is to *reuse off-policy* information from past iterations, such as previous *gradients* or *trajectories*. While gradient reuse has received substantial theoretical attention, leading to improved rates of $\mathcal{O}(\epsilon^{-3/2})$, the reuse of past trajectories remains largely unexplored from a theoretical perspective. In this work, we provide the first rigorous theoretical evidence that extensive reuse of past off-policy trajectories can significantly accelerate convergence in PG methods. We introduce a *power mean* correction to the *multiple importance weighting* estimator and propose RPG (Retrospective Policy Gradient), a PG algorithm that combines old and new trajectories for policy updates. Through a novel analysis, we show that, under established assumptions, RPG achieves a sample complexity of $\widetilde{\mathcal{O}}(\epsilon^{-1})$, the best known rate in the literature. We further validate empirically our approach against PG methods with state-of-the-art rates.

1 Introduction

Among *reinforcement learning* (RL, Sutton and Barto, 2018) approaches, *policy gradient* (PG, Deisenroth et al., 2013) methods have demonstrated notable success in tackling real-world problems, due to their capacity to operate in continuous state and action spaces (Peters and Schaal, 2006), robustness to sensor and actuator noise (Gravell et al., 2020), and effectiveness in managing partially observable environments (Azizzadenesheli et al., 2018). Moreover, PG methods allow for the integration of expert prior knowledge into policy design (Ghavamzadeh and Engel, 2006), thereby enhancing the safety, interpretability, and performance of the learned policy (Peters and Schaal, 2008). PG methods operate by directly optimizing the parameters $\boldsymbol{\theta} \in \mathbb{R}^{d_\Theta}$ of *parametric policies* to maximize a performance objective $J(\boldsymbol{\theta})$, typically the expected return. In practice, the parameter

Preprint. Under review.

vector $\boldsymbol{\theta}$ is updated via gradient ascent, based on an estimate of the gradient $\nabla J(\boldsymbol{\theta})$ w.r.t. the policy parameters. The goal is to identify an optimal parameterization $\boldsymbol{\theta}^*$ that maximizes the objective. In most cases $J(\boldsymbol{\theta})$ is a non-convex objective, thus this notion of optimality is often relaxed to that of finding a *first-order stationary point*, defined by the condition $\|\nabla J(\boldsymbol{\theta}^*)\|_2 = 0$ (Papini et al., 2018).

Despite their practical success, PG methods remain notoriously data-hungry, as each update requires several interactions with the environment to gather fresh data for gradient estimation. This limitation is reflected in their theoretical convergence guarantees, which have become a central focus in the PG literature. Vanilla PGs such as REINFORCE (Williams, 1992), PGT (Sutton et al., 1999), and GPOMDP (Baxter and Bartlett, 2001) aim to learn $\boldsymbol{\theta}^*$ via stochastic gradient ascent, using only *on-policy* trajectories, i.e., those generated by the current policy. These methods require order of $\mathcal{O}(\epsilon^{-2})$ total trajectories to reach an ϵ-approximate stationary point, i.e., a parameter $\boldsymbol{\theta}$ such that $\|\nabla J(\boldsymbol{\theta})\|_2^2 \leqslant \epsilon$. These unsatisfactory theoretical guarantees of vanilla PG methods are primarily due to the high variance of the gradient estimates, which are computed using only on-policy trajectories. To mitigate this variance, one can reuse *off-policy* data collected during the learning process, such as *past gradients* or *past trajectories* originating from different policy parameterizations. These data are typically incorporated into the gradient estimation through *importance weighting* (Owen and Zhou, 2000). A central challenge in this setting arises from the use of *importance weights* (IWs), which may inject high variance in the gradient estimate (Mandel et al., 2014).

While reusing trajectories is the most natural choice for improving PGs' sample efficiency, the literature extensively focused instead on reusing gradients, proposing various update schemes. Among these methods, Papini et al. (2018) introduced SVRPG, which incorporates ideas from stochastic variance-reduced gradient methods (Johnson and Zhang, 2013; Allen-Zhu and Hazan, 2016; Reddi et al., 2016). Specifically, SVRPG employs a *semi-stochastic gradient* that combines the stochastic gradient at the current iterate with that of a past "snapshot" parameterization. This method achieves a sample complexity of $\mathcal{O}(\epsilon^{-5/3})$ (Xu et al., 2020). An improvement over this result was proposed by Xu et al. (2019), who introduced SRVRPG. Unlike SVRPG, SRVRPG employs a *recursive semi-stochastic gradient*, which integrates the current stochastic gradient with those accumulated throughout the entire learning process. This recursive structure reduces the sample complexity to $\mathcal{O}(\epsilon^{-3/2})$ under the same convergence criterion. Furthermore, Yuan et al. (2020) proposed STORM-PG, which, instead of alternating between small and large batch updates as in SRVRPG, maintains a *moving average* of past stochastic gradients. This approach enables adaptive step sizes and eliminates the need for large batches of trajectories, while still ensuring a sample complexity of $\mathcal{O}(\epsilon^{-3/2})$. More recently, Paczolay et al. (2024) introduced a variant of STORM-PG that stochastically decides whether or not to reuse past gradients at each iteration. This method retains the same sample complexity of $\mathcal{O}(\epsilon^{-3/2})$, while relaxing the standard assumption of bounded IW variance. Beyond gradient reuse, variance reduction can also be achieved by reusing *off-policy* trajectories collected with previous policies. While this approach is conceptually more natural, it has received relatively little theoretical attention. For instance, Metelli et al. (2018) propose reusing trajectories to compute multiple successive stochastic gradient estimates, but with no formal convergence guarantees. Similarly, Papini et al. (2024) leverage past trajectories collected under multiple policy parameterizations, achieving a sample complexity of $\mathcal{O}(\epsilon^{-5/3})$, under strong technical assumptions. Table 1 summarizes the key assumptions, data reuse strategies, and sample complexities of related PG methods.

Original Contribution. Despite significant progress in reducing sample complexity, state-of-the-art methods still require $\mathcal{O}(\epsilon^{-3/2})$ trajectories to reach an ϵ-approximate stationary point. Most of these improvements rely on gradient reuse, a theoretically convenient but arguably less natural strategy than reusing trajectories. This observation raises a fundamental question: *Can the extensive reuse of past off-policy trajectories lead to provable improvements in convergence guarantees for PG methods?*

In this work, we answer this question affirmatively by introducing the RPG (Retrospective Policy Gradient) method that estimates the gradient direction via a *power mean* (PM) corrected version of the *multiple importance weighting* (MIW) estimator. The convergence guarantees of RPG are established exploiting the properties of the PM gradient estimator, which are derived through a novel analysis. Our main contributions are summarized as follows:

- In Section 3, we introduce the PM gradient estimator, a variant of MIW applying PM correction originally proposed in the single-IW setting by Metelli et al. (2021). We also present RPG that leverages this estimator to update the policy parameters. RPG fully exploits *all* previously collected off-policy trajectories, along with a *constant-size batch* of newly collected on-policy samples.

Algorithm	Data Reused	Assumptions		Sample Complexity
		Regular Policies[§]	IW Bounded Variance	
REINFORCE (Williams, 1992)	—	✓	✗	$\mathcal{O}\left(\epsilon^{-2}\right)$
PGT (Sutton et al., 1999)	—	✓	✗	$\mathcal{O}\left(\epsilon^{-2}\right)$
GPOMDP (Baxter and Bartlett, 2001)	—	✓	✗	$\mathcal{O}\left(\epsilon^{-2}\right)$
BPO (Papini et al., 2024)	Trajectories	✓ [†]	✓ [†]	$\mathcal{O}\left(\epsilon^{-5/3}\right)$
SVRPG (Papini et al., 2018; Xu et al., 2020)	Gradients	✓	✓	$\mathcal{O}\left(\epsilon^{-5/3}\right)$
SRVRPG (Xu et al., 2019)	Gradients	✓	✓	$\mathcal{O}\left(\epsilon^{-3/2}\right)$
STORM-PG (Yuan et al., 2020)	Gradients	✓	✓	$\mathcal{O}\left(\epsilon^{-3/2}\right)$
DEF-PG (Paczolay et al., 2024)	Gradients	✓	✗	$\mathcal{O}\left(\epsilon^{-3/2}\right)$
RPG (this work)	Trajectories	✓	✓	$\widetilde{\mathcal{O}}\left(\epsilon^{-1}\right)^{\#}$

[†] Stricter assumptions implying the ones matched. [#] $\widetilde{\mathcal{O}}(\cdot)$ hides logarithmic factors. [§] See Assumption 4.4.

Table 1: Comparison of the sample complexities of methods achieving $\|\nabla J(\boldsymbol{\theta})\|_2^2 \leqslant \epsilon$.

- In Section 4, we derive high-probability upper bounds on the estimation error of the PM estimator. These results are obtained by employing a martingale-based argument combined with the Freedman's inequality and a covering argument.
- In Section 5, we prove that RPG achieves a sample complexity of $\widetilde{\mathcal{O}}(\epsilon^{-1})$ for reaching an ϵ-approximate stationary point, under standard assumptions. This is made possible by a novel analysis that builds upon the results of Section 4, and provides the first rigorous theoretical evidence that extensive trajectory reuse can accelerate convergence in PG methods.

Comparative experimental results are reported in Section 6. All the proposed results extend to parameter-based PGs (Sehnke et al., 2010), whose discussion is deferred to Appendix A.

2 Background and Notation

Notation. For $n, m \in \mathbb{N}$ with $n \geqslant m$, we denote $[\![n]\!] := \{1, \ldots, n\}$ and $[\![m, n]\!] := \{m, \ldots, n\}$. For a measurable set $\mathcal{X}$, we denote with $\Delta(\mathcal{X})$ the set of probability measures over $\mathcal{X}$. For $P \in \Delta(\mathcal{X})$, we denote with p its density function w.r.t. a reference measure that we assume to exist whenever needed. For $P, Q \in \Delta(\mathcal{X})$, we denote that P is absolutely continuous w.r.t. Q as $P \ll Q$. If $P \ll Q$, the χ^2 divergence is defined as $\chi^2(P\|Q) := \left(\int_{\mathcal{X}} p(x)^2 q(x)^{-1} \mathrm{d}x\right) - 1$.

Lipschitz Continuous and Smooth Functions. A function $f : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ is L_1-*Lipschitz continuous* (L_1-LC) if $|f(\mathbf{x}) - f(\mathbf{x}')| \leqslant L_1 \|\mathbf{x} - \mathbf{x}'\|_2$ for every $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. Similarly, f is L_2-*Lipschitz smooth* (L_2-LS) if it is continuously differentiable and its gradient ∇f is L_2-LC, i.e., $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{x}')\|_2 \leqslant L_2 \|\mathbf{x} - \mathbf{x}'\|_2$ for every $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$.

Markov Decision Processes. A Markov Decision Process (MDP, Puterman, 1990) is represented by $\mathcal{M} := (\mathcal{S}, \mathcal{A}, p, r, \rho_0, \gamma)$, where $\mathcal{S} \subseteq \mathbb{R}^{d_{\mathcal{S}}}$ and $\mathcal{A} \subseteq \mathbb{R}^{d_{\mathcal{A}}}$ are the measurable state and action spaces; $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition model, where $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ specifies the probability density of landing in state $\mathbf{s}' \in \mathcal{S}$ by playing action $\mathbf{a} \in \mathcal{A}$ in state $\mathbf{s} \in \mathcal{S}$; $r : \mathcal{S} \times \mathcal{A} \to [-R_{\max}, R_{\max}]$ is the reward function, where $r(\mathbf{s}, \mathbf{a})$ specifies the reward the agent gets when playing action $\mathbf{a}$ in state $\mathbf{s}$; $\rho_0 \in \Delta(\mathcal{S})$ is the initial-state distribution; $\gamma \in [0, 1]$ is the discount factor. A trajectory $\tau = (\mathbf{s}_{\tau,0}, \mathbf{a}_{\tau,0}, \ldots, \mathbf{s}_{\tau,T-1}, \mathbf{a}_{\tau,T-1})$ of length $T \in \mathbb{N} \cup \{+\infty\}$ is a sequence of T state-action pairs. In the following, we refer to $\mathcal{T}$ as the set of all the possible trajectories. The *discounted return* of a trajectory $\tau \in \mathcal{T}$ is given by $R(\tau) := \sum_{t=0}^{T-1} \gamma^t r(\mathbf{s}_{\tau,t}, \mathbf{a}_{\tau,t})$. We admit $\gamma = 1$ only when $T < +\infty$.

Policy Gradients. Consider a *parametric stochastic policy* $\pi_{\boldsymbol{\theta}} : \mathcal{S} \to \Delta(\mathcal{A})$, where $\boldsymbol{\theta} \in \Theta$ is the parameter vector belonging to the parameter space $\Theta \subseteq \mathbb{R}^{d_\Theta}$. The policy is used to sample actions $\mathbf{a}_t \sim \pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}_t)$ to be played in state $\mathbf{s}_t$ for *every step* t of interaction. The performance of $\pi_{\boldsymbol{\theta}}$ is assessed via the *expected return* $J : \Theta \to \mathbb{R}$, defined as $J(\boldsymbol{\theta}) \coloneqq \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}}[R(\tau)]$, where $p_{\boldsymbol{\theta}}(\tau) \coloneqq \rho_0(\mathbf{s}_{\tau,0}) \prod_{t=0}^{T-1} \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,t}|\mathbf{s}_{\tau,t}) p(\mathbf{s}_{\tau,t+1}|\mathbf{s}_{\tau,t}, \mathbf{a}_{\tau,t})$ is the density function of trajectory τ induced by policy $\pi_{\boldsymbol{\theta}}$. The goal is to learn $\boldsymbol{\theta}^* \in \arg\max_{\boldsymbol{\theta} \in \Theta} J(\boldsymbol{\theta})$ and we denote $J^* \coloneqq J(\boldsymbol{\theta}^*)$.

On-Policy Estimators. If $J(\boldsymbol{\theta})$ is differentiable w.r.t. $\boldsymbol{\theta}$, PG methods (Peters and Schaal, 2008) update the parameter $\boldsymbol{\theta}$ via stochastic gradient ascent: $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \zeta_k \widehat{\nabla} J(\boldsymbol{\theta}_k)$, where $\zeta_k > 0$ is the *step size* and $\widehat{\nabla} J(\boldsymbol{\theta})$ is an estimator of $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. In particular, $\widehat{\nabla} J(\boldsymbol{\theta})$ often takes following form:

$$\widehat{\nabla} J(\boldsymbol{\theta}) = \frac{1}{N} \sum_{j=0}^{N-1} \mathbf{g}_{\boldsymbol{\theta}}(\tau_j),$$

being $\mathbf{g}_{\boldsymbol{\theta}}(\tau)$ a single-trajectory gradient estimator and N the number of independent trajectories $\{\tau_j\}_{j=0}^{N-1}$ collected with policy $\pi_{\boldsymbol{\theta}}$ (i.e., $\tau_j \sim p_{\boldsymbol{\theta}}$), called *batch size*. Classical on-policy unbiased gradient estimators are REINFORCE (Williams, 1992) and GPOMDP (Baxter and Bartlett, 2001), which are respectively defined as follows:

$$\mathbf{g}_{\boldsymbol{\theta}}^{\mathrm{R}}(\tau) = \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,t}|\mathbf{s}_{\tau,t}) R(\tau), \quad \mathbf{g}_{\boldsymbol{\theta}}^{\mathrm{G}}(\tau) = \sum_{t=0}^{T-1} \left(\sum_{l=0}^{t} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,l}|\mathbf{s}_{\tau,l}) \right) \gamma^t r(\mathbf{s}_{\tau,t}, \mathbf{a}_{\tau,t}).$$

In the following, we use $\mathbf{g}_{\boldsymbol{\theta}}(\tau)$ whenever it is possible to employ both the estimators.

Off-Policy Estimators. The gradient $\nabla J(\boldsymbol{\theta})$ can also be estimated by employing trajectories $\{\tau_j\}_{j=0}^{N-1}$ collected via a behavioral policy $\pi_{\boldsymbol{\theta}_b}$. In particular, under the assumption that $\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) \ll \pi_{\boldsymbol{\theta}_b}(\cdot|\mathbf{s})$ for every $\mathbf{s} \in \mathcal{S}$, the *(single) off-policy gradient estimator* (Owen, 2013) is defined as follows:

$$\widehat{\nabla}^{\mathrm{IS}} J(\boldsymbol{\theta}) = \frac{1}{N} \sum_{j=0}^{N-1} \frac{p_{\boldsymbol{\theta}}(\tau_j)}{p_{\boldsymbol{\theta}_b}(\tau_j)} \mathbf{g}_{\boldsymbol{\theta}}(\tau_j),$$

where $\tau_j \sim p_{\boldsymbol{\theta}_b}$ and $\frac{p_{\boldsymbol{\theta}}(\tau)}{p_{\boldsymbol{\theta}_b}(\tau)}$ is the Importance Weight (IW, Owen and Zhou, 2000) of the trajectory $\tau \in \mathcal{T}$, defined as $\frac{p_{\boldsymbol{\theta}}(\tau)}{p_{\boldsymbol{\theta}_b}(\tau)} = \prod_{t=0}^{T-1} \frac{\pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,t}|\mathbf{s}_{\tau,t})}{\pi_{\boldsymbol{\theta}_b}(\mathbf{a}_{\tau,t}|\mathbf{s}_{\tau,t})}$. We call $\widehat{\nabla}^{\mathrm{IW}} J(\boldsymbol{\theta})$ the Importance Weighting (IW) estimator, which is unbiased for a fixed $\boldsymbol{\theta} \in \Theta$ (i.e., non depending on the collected trajectories).The variance of the IW (Cortes et al., 2010) is related to the χ^2-divergence as shown in the following:

$$\underset{\tau \sim p_{\boldsymbol{\theta}_b}}{\mathbb{V}\mathrm{ar}} \left[\frac{p_{\boldsymbol{\theta}}(\tau)}{p_{\boldsymbol{\theta}_b}(\tau)} \right] = \chi^2(p_{\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}_b}).$$

This approach can be extended to take into account trajectories collected under different behavioral policies. Consider $m \in \mathbb{N}$ behavioral policies $\{\boldsymbol{\theta}_i\}_{i=0}^{m-1}$ and suppose to have collected N_i trajectories $\{\tau_{i,j}\}_{j=0}^{N_i-1}$ for each $\boldsymbol{\theta}_i$ (i.e., $\tau_{i,j} \sim p_{\boldsymbol{\theta}_i}$) with $i \in [\![0, m-1]\!]$. Let $\beta_i \geqslant 0$ be a *partition of the unit*, i.e., $\sum_{i=0}^{m-1} \beta_i(\tau) = 1$ for every $\tau \in \mathcal{T}$. Under the assumption that $\beta_i \pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) \ll \pi_{\boldsymbol{\theta}_i}(\cdot|\mathbf{s})$ the *multiple off-policy gradient estimator* (Veach and Guibas, 1995; Owen, 2013) is defined as:

$$\widehat{\nabla}^{\mathrm{MIW}} J(\boldsymbol{\theta}) = \sum_{i=0}^{m-1} \frac{1}{N_i} \sum_{j=0}^{N_i-1} \beta_i(\tau_{i,j}) \frac{p_{\boldsymbol{\theta}}(\tau_{i,j})}{p_{\boldsymbol{\theta}_i}(\tau_{i,j})} \mathbf{g}_{\boldsymbol{\theta}}(\tau_{i,j}), \tag{1}$$

Also in this case, the estimator is unbiased for a fixed $\boldsymbol{\theta} \in \Theta$. In the following, we refer to this gradient estimator as the Multiple Importance Weighting (MIW) one.

3 Trajectory Reuse in Policy Optimization

As highlighted in Section 1, vanilla PG methods rely exclusively on on-policy trajectories. A natural strategy to improve the sample efficiency is to *reuse off-policy trajectories* collected during previous iterations. However, since these trajectories are generated under different policy parameterizations, the resulting gradient estimates must be corrected accordingly through MIW (Owen, 2013).

In this section, we first formalize this learning scenario relying on the generic MIW gradient estimator presented in Equation (1). We then discuss why the Balance Heuristic (BH, Veach and Guibas,

1995), the most studied instantiation of the MIW estimator, is not well-suited for this setting. Next, we propose a PM-corrected version of the MIW estimator, referred to as the PM estimator, which addresses the limitations of the BH one. Finally, we introduce RPG (Retrospective Policy Gradient), a PG algorithm that leverages the PM estimator to perform parameter updates.

Learning Scenario. Consider a generic PG method that, at the k^{th} iteration, collects a constant-size batch of N trajectories $\{\tau_{k-1,j}\}_{j=0}^{N-1}$ using the current policy parameterization $\boldsymbol{\theta}_{k-1}$ (i.e., $\tau_{k-1,j} \sim p_{\boldsymbol{\theta}_{k-1}}$). The policy is then updated via gradient ascent: $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_{k-1} + \zeta_{k-1}\widehat{\nabla}^{\text{MIW}}J(\boldsymbol{\theta}_{k-1})$, where ζ_{k-1} is the step size and $\widehat{\nabla}^{\text{MIW}}J$ denotes a MIW gradient estimator, as defined in Equation (1). The estimator $\widehat{\nabla}^{\text{MIW}}J$ leverages the full set of collected trajectories $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$, where each $\tau_{i,j} \sim p_{\boldsymbol{\theta}_i}$, being $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ the set of policy parameters encountered up to iteration k. Note that, because of the gradient ascent update, the parameters $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ are all *statistically dependent* since each $\boldsymbol{\theta}_i$ depends on the past collected trajectories $\{\{\tau_{l,j}\}_{j=0}^{N-1}\}_{l=0}^{i-1}$ used for the gradient step.

MIW with Balance Heuristic. The most studied choice for the coefficients β_i of the MIW estimator from Equation (1) is the BH (Veach and Guibas, 1995). In our learning scenario they take the form $\beta_i^{\text{BH}}(\tau) := \frac{p_{\boldsymbol{\theta}_i}(\tau)}{\sum_{l=0}^{k-1} p_{\boldsymbol{\theta}_l}(\tau)}$ for every $\tau \in \mathcal{T}$, leading to the BH estimator:

$$\widehat{\nabla}^{\text{BH}}J(\boldsymbol{\theta}_{k-1}) = \frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{p_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j})}{\sum_{l=0}^{k-1} p_{\boldsymbol{\theta}_l}(\tau_{i,j})}\mathbf{g}_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j}).$$

The BH estimator enjoys the *defensive* property, i.e., the IWs are *bounded* by k. Moreover, *when the behavioral policies $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ are statistically independent*, which is not the case of our learning scenario, the BH estimator is proven to enjoy nearly-optimal variance (Veach and Guibas, 1995, Theorem 1). However, in our learning scenario, the BH estimator suffers from both practical and theoretical limitations. Specifically, computing each IW requires evaluating the likelihood of each trajectory $\tau_{i,j}$ under *all* policies. More critically, the BH requires evaluating both older trajectories under newer policies and *newer trajectories under older policies*, as shown in the following:

$$\widehat{\nabla}^{\text{BH}}J(\boldsymbol{\theta}_{k-1}) = \frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{p_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j})}{\sum_{l=0}^{i} p_{\boldsymbol{\theta}_l}(\tau_{i,j}) + \sum_{l=i+1}^{k-1} p_{\boldsymbol{\theta}_l}(\tau_{i,j})}\mathbf{g}_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j}).$$

The part in (–) illustrates how the BH introduces circular dependency among the random variables, violating the natural temporal ordering, with the effect of introducing *bias* into the estimate. Consider, for instance, the random variable $p_{\boldsymbol{\theta}_l}(\tau_{i,j})$, with $l < i$: it depends on the trajectory $\tau_{i,j} \sim p_{\boldsymbol{\theta}_i}$, while the parameterization $\boldsymbol{\theta}_i$ itself is a random variable computed from the earlier parameterization $\boldsymbol{\theta}_l$ and its associated trajectories $\tau_{l,j} \sim p_{\boldsymbol{\theta}_l}$. Notably, we cannot simply omit the problematic terms from the summation, as this would introduce additional bias, since the resulting coefficients would be no longer a partition of the unit. A graphical representation of these issues is provided in Appendix B.

Power Mean-corrected MIW. To overcome the limitations of the BH estimator, we could select the coefficients as $\beta_i(\tau) := \alpha_{i,k} \geqslant 0$ for all $i \in [\![0, k-1]\!]$, making them depend only on the parameterization $\boldsymbol{\theta}_i$ and the total number of parameterizations k, with the constraint $\sum_{i=0}^{k-1}\alpha_{i,k} = 1$. While this choice eliminates the bias introduced by the violation of the natural temporal ordering, it no longer guarantees the defensive property. To overcome also this issue, we borrow the idea of a PM-correction of the IW, which was proposed for the single-IW setting by Metelli et al. (2021), and extend it to the MIW estimator, leading to the following:

$$\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_{k-1}) = \frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{\alpha_{i,k}p_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j})}{(1-\lambda_{i,k})p_{\boldsymbol{\theta}_i}(\tau_{i,j}) + \lambda_{i,k}p_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j})}\mathbf{g}_{\boldsymbol{\theta}_{k-1}}(\tau_{i,j}),$$

with $\lambda_{i,k} \in [0, 1]$ for every $i \in [\![0, k-1]\!]$. The PM correction computes a weighted power mean with exponent -1 between the vanilla IW and 1. Importantly, whenever $\lambda_{i,k} > 0$, the PM-corrected IW is bounded by $\alpha_{i,k}/\lambda_{i,k}$. We stress that the PM estimator offers significant practical advantages, overcoming the reported pitfalls of the BH one. Specifically, this estimator requires evaluating the likelihood of each trajectory $\tau_{i,j}$ only w.r.t. the policy under which it was collected and the current target policy $\boldsymbol{\theta}_{k-1}$. This eliminates both the computational inefficiency and the temporal inconsistencies of the BH estimator, where newer trajectories must be evaluated under older policies.

Algorithm 1: RPG.

Input : Iterations K, Batch Size N, Learning Rate Schedule $\{\zeta_k\}_{k=0}^{K-1}$, Initial Parameterization $\boldsymbol{\theta}_0$.
for $k \in [\![0, K-1]\!]$ **do**
  Collect N trajectories $\{\tau_{k,j}\}_{j=0}^{N-1}$ with policy $\pi_{\boldsymbol{\theta}_k}$.
  Update the policy parameterization: $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \zeta_k \widehat{\nabla}^{\mathrm{PM}} J(\boldsymbol{\theta}_k)$.
end
Return $\boldsymbol{\theta}_{\mathrm{OUT}} \in \{\boldsymbol{\theta}_i\}_{i=0}^{K-1}$ chosen uniformly at random.

Clearly, the presence of the correcting term $\lambda_{i,k}$ introduces a *bias* that, differently from that of BH, is easily manageable. The reported properties make the PM estimator particularly well-suited for the described learning scenario. These advantages are graphically represented in Appendix B.

The RPG Method. Having introduced the PM estimator, our proposed method RPG, whose pseudo-code is provided in Algorithm 1, follows the learning scenario described earlier in this section, but replaces the generic MIW estimator with the PM one.

4 PM Estimator: Dealing with the Estimation Error

In this section, we provide high-probability upper bounds on the estimation error $\|\widehat{\nabla}^{\mathrm{PM}} J(\boldsymbol{\theta}) - \nabla J(\boldsymbol{\theta})\|_2$ when using the PM estimator, setting the basis for the convergence analysis of RPG.

Before presenting the results, we highlight a key difficulty in the analysis. As already mentioned, the target parameterization $\boldsymbol{\theta}_{k-1}$ is the output of a stochastic process and depends on the previously collected trajectories $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-2}$. As a consequence, the PM estimator $\widehat{\nabla}^{\mathrm{PM}} J(\boldsymbol{\theta}_{k-1})$ of $\nabla J(\boldsymbol{\theta}_{k-1})$ cannot be easily analyzed using standard martingale-based concentration bounds due to such *statistical dependence*. To address this, as an intermediate step, we first derive an upper bound for the case where the target parameterization is *independent of the history*. Then, we extend the result to the real scenario in which $\boldsymbol{\theta}_{k-1}$ is the actual *outcome of the learning process*.

4.1 Bounding the PM Estimation Error for a Fixed Target Parameterization

Before stating the result, we introduce the two following assumptions.

Assumption 4.1 (Bounded Single-Trajectory On-Policy Gradient Estimator)**.** *There exist $G < +\infty$ and $G_2 < +\infty$ such that the single-trajectory on-policy gradient estimator $\boldsymbol{g}_{\boldsymbol{\theta}}(\tau)$ enjoys:* $\sup_{\boldsymbol{\theta},\tau \in \Theta \times \mathcal{T}} \|\boldsymbol{g}_{\boldsymbol{\theta}}(\tau)\|_2 \leqslant G$ *and* $\sup_{\boldsymbol{\theta},\tau \in \Theta \times \mathcal{T}} \|\nabla \boldsymbol{g}_{\boldsymbol{\theta}}(\tau)\|_2 \leqslant G_2$.

Assumption 4.1 is satisfied by both $\mathbf{g}_{\boldsymbol{\theta}}^{\mathrm{R}}(\tau)$ and $\mathbf{g}_{\boldsymbol{\theta}}^{\mathrm{G}}(\tau)$ whenever $\|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})\|_2$ and $\|\nabla_{\boldsymbol{\theta}}^2 \log \pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})\|_2$ are bounded for every $\mathbf{a} \in \mathcal{A}$ and $\mathbf{s} \in \mathcal{S}$ (see Lemma C.2), which are common assumptions in the policy gradient literature (Papini et al., 2018; Xu et al., 2019; Yuan et al., 2020). In the following, we consider a generic single-trajectory on-policy estimator.

Assumption 4.2 (Bounded χ^2 Divergence)**.** *There exists a known constant $D \in \mathbb{R}_{\geqslant 1}$ such that:* $\sup_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2 \in \Theta} \chi^2\left(p_{\boldsymbol{\theta}_1} \| p_{\boldsymbol{\theta}_2}\right) \leqslant D - 1$.

Assumption 4.2 enforces that the variance of the vanilla IWs is bounded, a standard assumption in the analysis of variance-reduced PG methods (Papini et al., 2018; Xu et al., 2019, 2020; Yuan et al., 2020). Moreover, as shown by Cortes et al. (2010), this assumption holds in the case of univariate Gaussian policies with $\sigma_1 < 2\sigma_2$, being σ_1 and σ_2 the standard deviations of $\pi_{\boldsymbol{\theta}_1}$ and $\pi_{\boldsymbol{\theta}_2}$ respectively. This is the central assumption for our theoretical results, as it enables a construction of the coefficients $\alpha_{i,k}$ and $\lambda_{i,k}$, appearing in the PM estimator, allowing us to derive strong bounds on the estimation error.

We are now ready to provide the first upper bound on the PM estimation error when considering a fixed target parameterization.

Theorem 4.1 (Fixed Target PM Estimation Error Bound)**.** *Consider to run RPG for k iterations, collecting the parameterizations $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ with trajectories $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$. Let $\bar{\boldsymbol{\theta}} \in \Theta$ be chosen*

independently on $\{\boldsymbol{\theta}_i, \{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$. *Under Assumptions 4.1 and 4.2, using the PM estimator with*

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log 6 + 4\log\frac{1}{\delta}}{3D_i Nk}} \quad \textit{and} \quad \alpha_{i,k} = \frac{D_i^{-1/2}}{\sum_{l=0}^{k-1} D_l^{-1/2}},$$

where $D_{k-1} \coloneqq 1$ *and* $D_i \coloneqq D$ *for* $i \in [\![0, k-2]\!]$, *for every* $\delta \in [0,1]$, *with probability at least* $1-\delta$, *it holds that:*

$$\left\| \widehat{\nabla}^{PM} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}}) \right\|_2 \leqslant 8G\sqrt{\frac{Dd_\Theta \log 6 + D\log\left(\frac{1}{\delta}\right)}{Nk}}.$$

Several comments are in order. First, the proposed upper bound on the PM estimation error depends on the parameter dimensionality d_Θ which is due to the Euclidean norm $\|\cdot\|_2$. It also depends on the constant D from Assumption 4.2. For interested readers, a tighter version of this bound, which reduces to the on-policy case where $D = +\infty$, is provided in the proof of Theorem 4.1. More importantly, we highlight that the PM estimation error scales as $\mathcal{O}((Nk)^{-1/2})$, where Nk is the *total number of trajectories* collected by RPG up to iteration k. It is worth noting that, instead, the standard on-policy estimators enjoy a concentration bound scaling with $\mathcal{O}(N^{-1/2})$ only (Papini et al., 2022), i.e., depending on the *batch size* only. This plays a key role in our sample complexity analysis. We also note that the construction of the coefficients $\alpha_{i,k}$ and $\lambda_{i,k}$ requires knowledge of the constant D (Assumption 4.2) and of the confidence δ, still leading to deterministic coefficients.

4.2 Bounding the PM Estimation Error in the Full Learning Process

Here, we extend the result of Theorem 4.1 to account for the *full learning process* of RPG where the target parameterization $\boldsymbol{\theta}_{k-1}$ is itself the outcome of k iterations of the learning algorithm.

Strategy Outline. To extend the result of Theorem 4.1 to this setting, we consider that the target parameterization $\boldsymbol{\theta}_{k-1}$ lies within a d_Θ-dimensional ball $\mathcal{B}_\rho^{d_\Theta} \subseteq \mathbb{R}^{d_\Theta}$ of radius ρ, chosen to ensure that all iterates remain inside the ball almost surely. We then apply a standard *covering argument* over $\mathcal{B}_\rho^{d_\Theta}$ to derive the desired *uniform bound* on the PM estimation error. To carry out this argument, we first need to introduce the following assumptions.

Assumption 4.3 (Smoothness of J)**.** *There exists* $L_{2,J} \in \mathbb{R}_{>0}$ *such that, for every* $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Theta$*:*

$$\|\nabla J(\boldsymbol{\theta}_1) - \nabla J(\boldsymbol{\theta}_2)\|_2 \leqslant L_{2,J} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 .$$

Assumption 4.4 (Regularity of $\log \pi_{\boldsymbol{\theta}}$)**.** *There exist* $L_{1,\pi}, L_{2,\pi} \in \mathbb{R}_{>0}$ *such that, for every* $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Theta$ *and for any* $\mathbf{a} \in \mathcal{A}$ *and* $\mathbf{s} \in \mathcal{S}$*:*

$$\|\log \pi_{\boldsymbol{\theta}_1}(\mathbf{a}|\mathbf{s}) - \log \pi_{\boldsymbol{\theta}_2}(\mathbf{a}|\mathbf{s})\|_2 \leqslant L_{1,\pi} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 , \tag{2}$$

$$\|\nabla \log \pi_{\boldsymbol{\theta}_1}(\mathbf{a}|\mathbf{s}) - \nabla \log \pi_{\boldsymbol{\theta}_2}(\mathbf{a}|\mathbf{s})\|_2 \leqslant L_{2,\pi} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 . \tag{3}$$

It is worth noting that these are common assumptions in both the standard PG literature (Williams, 1992; Sutton et al., 1999; Baxter and Bartlett, 2001) and the variance-reduced PG one (Papini et al., 2018; Xu et al., 2019, 2020; Yuan et al., 2020).

We are now ready to extend the result of Theorem 4.1 to the setting in which the target parameterization for $\widehat{\nabla}^{\text{PM}} J(\cdot)$ results from the stochastic learning process of RPG.

Theorem 4.2. *Consider to run RPG for* k *iterations with a constant step size* ζ*, collecting the parameterizations* $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ *with trajectories* $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$. *Under Assumptions 4.1, 4.2, 4.3, and 4.4, select the* $\alpha_{i,k}$ *terms as in Theorem 4.1 and, for every* $\delta \in [0,1]$, *the* $\lambda_{i,k}$ *terms as:*

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left((18\sqrt{3}L_{2,J} + 27L_1\sqrt{DNk})\frac{\zeta Nk^2}{16\delta\sqrt{d_\Theta}}\right) + 4\log\frac{1}{\delta}}{3D_i Nk}},$$

where $L_1 \coloneqq GTL_{1,\pi} + G_2$. *With probability at least* $1-\delta$, *it holds:*

$$\left\| \widehat{\nabla}^{PM} J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1}) \right\|_2 \leqslant 16G\sqrt{\frac{Dd_\Theta}{Nk}\log\left(6\zeta\left(L_{2,J} + L_1\sqrt{\frac{3}{4}DNk}\right)\frac{Nk^2}{\delta\sqrt{d_\Theta}}\right)}.$$

Note that by extending the result of Theorem 4.1 to the setting where the target parameterization is generated by the learning process, the PM estimation error remains of order $\widetilde{\mathcal{O}}((Nk)^{-1/2})$, i.e., scaling with the total trajectories Nk collected up to iteration k. Thus, compared to Theorem 4.1, this extension introduces only additional logarithmic factors, preserving the dependencies on d_θ and D. In the next section, we leverage Theorem 4.2 to establish sample complexity guarantees for RPG.

5 RPG: Sample Complexity

Equipped with our bound on the PM estimation error for a stochastic target from Theorem 4.2, we are ready to study the sample complexity of RPG to converge to an ϵ-approximate stationary point.

Theorem 5.1 (RPG Sample Complexity)**.** *Consider to run RPG for $K \in \mathbb{N}$ iterations. Under Assumptions 4.1, 4.2, 4.3, and 4.4, for every $k \in [\![K]\!]$ select the terms $\alpha_{i,k}$ as in Theorem 4.1 and the terms $\lambda_{i,k}$ as:*

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left((18\sqrt{3}L_{2,J} + 27L_1\sqrt{DNk})\frac{\zeta G^2 NKk^2}{8\epsilon\sqrt{d_\Theta}}\right) + 4\log\frac{2G^2K}{\epsilon}}{3D_i Nk}},$$

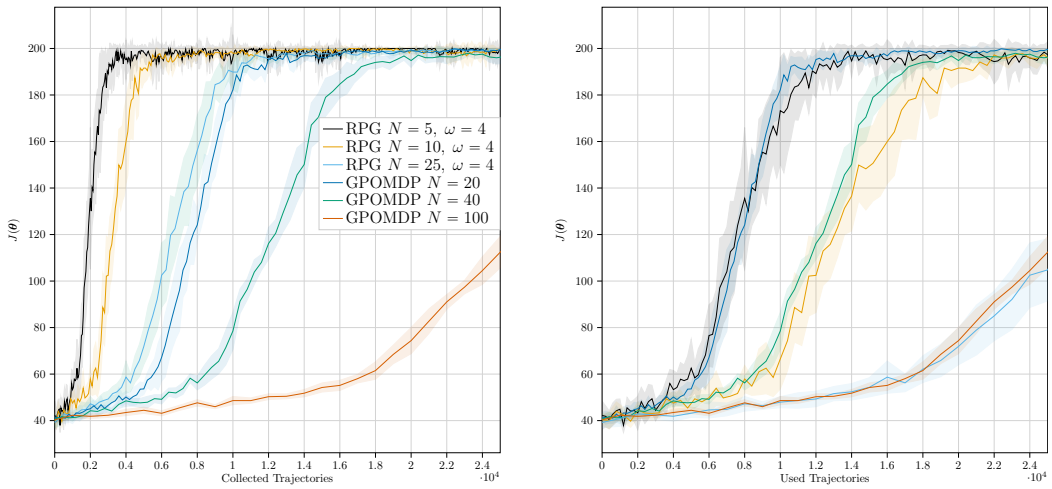
where $L_1 := GTL_{1,\pi} + G_2$. Selecting a constant step size $\zeta \leqslant 1/L_{2,J}$, with a sample complexity $NK \geqslant \widetilde{\mathcal{O}}(G^2 D d_\Theta \epsilon^{-1})$ and an iteration complexity $K \geqslant \mathcal{O}(\epsilon^{-1})$, it is guaranteed that $\mathbb{E}[\|\nabla J(\boldsymbol{\theta}_{OUT})\|_2^2] \leqslant \epsilon$, where the expectation is taken w.r.t. the learning process and the uniform sampling of $\boldsymbol{\theta}_{OUT}$ from $\{\boldsymbol{\theta}_k\}_{k=0}^{K-1}$.

We emphasize that the result presented in Theorem 5.1 is obtained by leveraging Theorem 4.2. This approach first yields a high-probability bound for reaching an ϵ-approximate stationary point, as detailed in the proof of Theorem 5.1. We then convert this high-probability bound into an expectation result. Additionally, we note that Theorem 5.1 establishes an *average-iterate* convergence result, as $\boldsymbol{\theta}_{\text{OUT}}$ is selected uniformly at random from the set of iterates $\{\boldsymbol{\theta}_k\}_{k=0}^{K-1}$, as common in this context (Papini et al., 2018; Xu et al., 2019; Yuan et al., 2020).

Strengths. As discussed in Section 1 and summarized in Table 1, under standard assumptions, RPG achieves a sample complexity of order $\widetilde{\mathcal{O}}(\epsilon^{-1})$, the best known rate in the policy gradient literature so far for stochastic gradients and general policy classes. This result is achieved by our *novel theoretical analysis* leveraging the properties of the PM estimator, and provides the first rigorous theoretical evidence of the benefits of reusing *all* trajectories collected throughout the learning process. Notably, our convergence guarantees are obtained under a *constant* batch size N (that can be even 1), in contrast to prior works (Papini et al., 2018; Xu et al., 2019, 2020; Yuan et al., 2020).

Potential Improvements. The current theoretical guarantees rely on selecting the PM estimator's coefficients $\alpha_{i,k}$ and $\lambda_{i,k}$ based on the knowledge of the constant D from Assumption 4.2. In practice, it would be desirable to adopt adaptive coefficients that depend not on D, but rather on the actual divergences among trajectory distributions induced by different policy parameterizations. However, this improvement would introduce additional stochasticity, significantly complicating the theoretical analysis and likely requiring a fundamentally different approach. Additionally, the current convergence rate exhibits a dependency on the parameter dimension d_Θ, differently to standard stochastic gradient methods. Finally, RPG requires access to *all* previously collected trajectories, which may be computationally impractical. To address this, in Section 6, we introduce a practical variant reusing only the ω most recent batches.

Comparison with the Existing Lower Bound. Paczolay et al. (2024) adapted a lower bound by Arjevani et al. (2023) for first-order non-convex stochastic optimization. They show that, with no assumption on the variance of the IWs, actor-only policy gradient algorithms need $\Omega(\epsilon^{-3/2})$ trajectories to find an ϵ-approximate stationary point in the worst case. It may seem that our $\widetilde{O}(\epsilon^{-1})$ upper bound for RPG contradicts the lower bound. However, the results are not directly comparable. First of all, the policy-class construction used in Paczolay et al. (2024) requires a large number of parameters $d_\Theta = \widetilde{O}(\epsilon^{-1})$. This is apparent in Theorem 3 by Arjevani et al. (2023), on which the lower bound of Paczolay et al. (2024) is based. While SGD/REINFORCE is *dimension-free*, i.e., its sample complexity upper bound does not explicitly depend on d_Θ (though this dependence may be implicitly hidden in the estimator's variance in the worst case), the sample complexity we establish



(a) Collected Trajectories.

(b) Used Trajectories.

Figure 1: *Cart Pole*. 10 runs (mean $\pm 95\%$ C.I.).

for RPG is $\widetilde{\mathcal{O}}(d_\Theta \epsilon^{-1})$. This translates into a $\widetilde{O}(\epsilon^{-2})$ sample complexity in the hard instance used to prove the $\Omega(\epsilon^{-3/2})$ lower bound, so there is no contradiction. Moreover, the lower bound does not require the variance of IWs to be bounded. Hence, we cannot exclude the existence of algorithms able to achieve dimension-free $\widetilde{O}(D\epsilon^{-1})$ sample complexity under Assumption 4.2.

6 Experiments

We now numerically validate our method. For the experimental campaign, we adapt RPG to reuse only the trajectories collected over the ω most recent iterates, rather than all trajectories collected since the beginning of training. We refer to ω as the *window size*. This modification is necessary to maintain computational tractability, as computing IWs and gradient estimates over the entire trajectory history would become increasingly expensive. Moreover, as training progresses, older parameterizations tend to diverge from the current one, causing the corresponding trajectories to receive lower IWs and contribute less to the gradient estimate. In addition, instead of selecting the $\alpha_{i,k}$ and $\lambda_{i,k}$ coefficients as prescribed in Theorem 5.1, we allow them to be dynamic. Since the constant D in Assumption 4.2 is generally unknown in practice, we estimate the divergence terms D_i between $p_{\boldsymbol{\theta}_i}$ and $p_{\boldsymbol{\theta}_{k-1}}$, where $\boldsymbol{\theta}_{k-1}$ is the current target parameterization and $i \in [\![\max\{0, k-\omega+1\}, k-1]\!]$. Implementation and experimental details, along with additional experiments, are provided in Appendices E and F.

On Reusing Trajectories. Figure 1 compares different configurations of RPG and GPOMDP in the *Cart Pole* environment (Barto et al., 1983), matching the total number of trajectories used to compute gradient estimates. Specifically, RPG was run with $\omega = 4$ and $N \in \{5, 10, 25\}$, while GPOMDP used $N \in \{20, 40, 100\}$. Both methods employ linear Gaussian policies with fixed variance $\sigma^2 = 0.3$ and were trained using the Adam optimizer (Kingma and Ba, 2015) with initial learning rate $\zeta_0 = 10^{-2}$. Figure 1b reports the performance $J(\boldsymbol{\theta})$ as a function of *training iterations*. The learning curves of RPG and GPOMDP closely align across matched configurations, empirically validating that, in this setting, reusing past trajectories provides nearly the same gradient information as collecting fresh data. However, each configuration of GPOMDP requires a larger number of environment interactions per update compared to its RPG counterpart. Figure 1a shows the same $J(\boldsymbol{\theta})$ plotted against the total number of *collected trajectories*. Here, the data efficiency of RPG becomes evident: for every matched configuration, RPG reaches optimal performance faster and with fewer environment interactions. These findings confirm that trajectory reuse enables both faster convergence and improved sample efficiency compared to relying solely on newly collected data.

Baselines Comparison. Figure 2 compares RPG with GPOMDP (not reusing trajectories) and several baselines with state-of-the-art rates: SVRPG, SRVRPG, STORM-PG, and DEF-PG, all discussed in Section 1. The experiment was conducted in the *Half Cheetah-v4* environment from the MuJoCo control suite (Todorov et al., 2012). All methods employ a 32×32 deep Gaussian policy with `tanh`

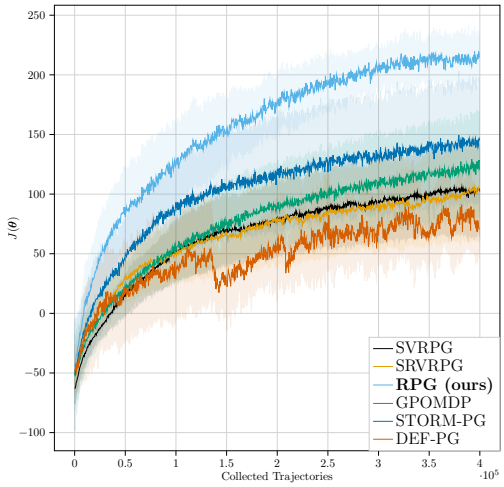


Figure 2: *Half Cheetah*. 10 runs (mean $\pm 95\%$ C.I.).

activations and fixed variance $\sigma^2 = 0.1$, and are trained using the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of $\zeta_0 = 10^{-4}$. RPG was run with $\omega = 8$ and $N = 40$. All other methods were configured to observe, on average, the same number of new trajectories per iteration, specifically 40, so as to ensure a fair comparison in terms of data collection. This averaging is particularly important for methods like SVRPG, SRVRPG, and DEF-PG, which alternate between large-batch snapshot gradients and mini-batch updates. As shown in Figure 2, RPG consistently outperforms all baselines under this setting, achieving nearly twice the final performance of all competing methods, despite all of them receive the same amount of new information per iteration. These results reinforce the conclusion that exploiting past experiences not only improves sample efficiency, but also accelerates convergence, potentially aiding in escaping local optima and achieving higher final performance. In Appendix F, we compare against these baselines in other environments.

7 Conclusion

In this work, we provide the first rigorous theoretical evidence that extensive reuse of trajectories collected during previous iterations can accelerate convergence in PGs. Specifically, we introduce RPG, a PG algorithm that leverages a PM-corrected version of the MIS estimator, and show that it achieves a rate of $\widetilde{\mathcal{O}}(\epsilon^{-1})$ to find an ϵ-accurate stationary point, the best-known sample complexity for this setting. Our theoretical guarantees are enabled by a novel analysis of the PM estimator's estimation error, but several directions remain open for improvement. First, RPG requires storing all past trajectories, which may be impractical; the memory-efficient variant with finite storage should be studied also from a theoretical perspective. Second, extending the analysis to support dynamic or stochastic PM coefficients, removing the reliance on the knowledge of D from Assumption 4.2, would be an important step forward. Additionally, removing this assumption entirely, as in (Paczolay et al., 2024), is also a worthwhile direction to explore. Finally, eliminating the dependency on d_Θ in the complexity bound and establishing global last-iterate convergence results (e.g., Fatkhullin et al., 2023; Montenegro et al., 2024) are valuable directions for future work.

References

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

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

Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *IEEE International Conference on Intelligent Robots and Systems*, pages 2219–2225. IEEE, 2006.

Benjamin Gravell, Peyman Mohajerin Esfahani, and Tyler Summers. Learning optimal controllers for linear systems with multiplicative noise via policy gradient. *IEEE Transactions on Automatic Control*, 66(11):5283–5298, 2020.

Kamyar Azizzadenesheli, Yisong Yue, and Animashree Anandkumar. Policy gradient in partially observable environments: Approximation and convergence. *arXiv preprint arXiv:1810.07900*, 2018.

Mohammad Ghavamzadeh and Yaakov Engel. Bayesian policy gradient algorithms. *Advances in Neural Information Processing Systems (NeurIPS)*, 19, 2006.

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

Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *International conference on machine learning*, pages 4026–4035. PMLR, 2018.

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

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Jonathan Baxter and Peter L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

Art Owen and Yi Zhou. Safe and effective importance sampling. *Journal of the American Statistical Association*, 95(449):135–143, 2000.

Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *AAMAS*, volume 1077, 2014.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pages 699–707. PMLR, 2016.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323. PMLR, 2016.

Pan Xu, Felicia Gao, and Quanquan Gu. An improved convergence analysis of stochastic variance-reduced policy gradient. In *Uncertainty in Artificial Intelligence*, pages 541–551. PMLR, 2020.

Pan Xu, Felicia Gao, and Quanquan Gu. Sample efficient policy gradient methods with recursive variance reduction. *arXiv preprint arXiv:1909.08610*, 2019.

Huizhuo Yuan, Xiangru Lian, Ji Liu, and Yuren Zhou. Stochastic recursive momentum for policy gradient methods. *arXiv preprint arXiv:2003.04302*, 2020.

Gabor Paczolay, Matteo Papini, Alberto Maria Metelli, Istvan Harmati, and Marcello Restelli. Sample complexity of variance-reduced policy gradient: weaker assumptions and lower bounds. *Machine Learning*, 113(9):6475–6510, 2024.

Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. *Advances in Neural Information Processing Systems*, 31, 2018.

Matteo Papini, Giorgio Manganini, Alberto Maria Metelli, and Marcello Restelli. Policy gradient with active importance sampling. *arXiv preprint arXiv:2405.05630*, 2024.

Alberto Maria Metelli, Alessio Russo, and Marcello Restelli. Subgaussian and differentiable importance sampling for off-policy evaluation and learning. *Advances in neural information processing systems*, 34:8119–8132, 2021.

Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010. International Conference on Artificial Neural Networks (ICANN).

Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

Art B. Owen. *Monte Carlo Theory, Methods and Examples*. Art Owen, 2013.

Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. *Advances in neural information processing systems*, 23, 2010.

Eric Veach and Leonidas J Guibas. Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 419–428, 1995.

Matteo Papini, Matteo Pirotta, and Marcello Restelli. Smoothing policies and safe policy gradients. *Machine Learning*, 111(11):4081–4137, 2022.

Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1): 165–214, 2023.

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

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015*, 2015.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

Ilyas Fatkhullin, Anas Barakat, Anastasia Kireeva, and Niao He. Stochastic policy gradient methods: Improved sample complexity for fisher-non-degenerate policies. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 9827–9869. PMLR, 2023.

Alessandro Montenegro, Marco Mussi, Alberto Maria Metelli, and Matteo Papini. Learning optimal deterministic policies with stochastic policy gradients. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 36160–36211. PMLR, 2024.

Alberto Maria Metelli, Matteo Papini, Nico Montali, and Marcello Restelli. Importance sampling techniques for policy optimization. *Journal of Machine Learning Research*, 21(141):1–75, 2020.

David A Freedman. On tail probabilities for martingales. *the Annals of Probability*, pages 100–118, 1975.

Rui Yuan, Robert M Gower, and Alessandro Lazaric. A general sample complexity analysis of vanilla policy gradient. In *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3332–3380. PMLR, 2022.

M. Gil, F. Alajaji, and T. Linder. Rényi divergence measures for commonly used univariate continuous distributions. *Information Sciences*, 249:124–131, 11 2013.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pages 1329–1338. PMLR, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1889–1897. JMLR.org, 2015.

Long-Ji Lin. *Reinforcement learning for robots using neural networks*. Carnegie Mellon University, 1992.

A On Employing Parameter-based Exploration in RPG

As always in RL, addressing the *exploration* problem is essential. This refers to the need for an agent to try out different actions, not necessarily to collect immediate rewards, but to gather information about the possible outcomes and long-term effects of actions. In PG methods, this is often done by carrying out the exploration either at the actions level or directly at the policy parameters level. These two exploration strategies are known as *action-based* (AB) and *parameter-based* (PB) exploration (Metelli et al., 2018; Montenegro et al., 2024), respectively. In particular, AB exploration, whose prototypical algorithms are REINFORCE (Williams, 1992) and GPOMDP (Baxter and Bartlett, 2001), keeps the exploration at the action level by leveraging *stochastic policies* (e.g., Gaussian). Instead, PB approaches, whose prototype is PGPE (Sehnke et al., 2010), explore at the parameter level via *stochastic hyperpolicies*, used to sample the parameters of an underlying (typically deterministic) policy.

For readability purposes, in the main paper we focused only on AB PG methods, while here we provide insights on the fact that all the proposed analysis works for PB PG methods as well.

A.1 Parameter-based Exploration

In PB exploration, we use a *parametric stochastic hyperpolicy* $\nu_{\boldsymbol{\xi}} \in \Delta(\Theta)$, where $\boldsymbol{\xi} \in \Xi \subseteq \mathbb{R}^{d_\Xi}$ is the hyperparameter vector. The hyperpolicy is used to sample parameters $\boldsymbol{\theta} \sim \nu_{\boldsymbol{\xi}}$ to be plugged in the underlying parametric policy $\pi_{\boldsymbol{\theta}}$ (that may also be deterministic) at the beginning of *every trajectory*. The performance index of $\nu_{\boldsymbol{\xi}}$ is $J_{\mathrm{P}} : \Xi \to \mathbb{R}$, that is the expectation over $\boldsymbol{\theta}$ of $J(\boldsymbol{\theta})$ defined as:

$$J_{\mathrm{P}}(\boldsymbol{\xi}) \coloneqq \mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim\nu_{\boldsymbol{\xi}}} \left[J(\boldsymbol{\theta}) \right]. \tag{4}$$

PB exploration aims at learning $\boldsymbol{\xi}^* \in \arg\max_{\boldsymbol{\xi}\in\Xi} J_{\mathrm{P}}(\boldsymbol{\xi})$ and we denote $J_{\mathrm{P}}^* \coloneqq J_{\mathrm{P}}(\boldsymbol{\xi}^*)$. If $J_{\mathrm{P}}(\boldsymbol{\xi})$ is differentiable w.r.t. $\boldsymbol{\xi}$, PGPE (Sehnke et al., 2010) updates the hyperparameter $\boldsymbol{\xi}$ via gradient ascent: $\boldsymbol{\xi}_{k+1} \leftarrow \boldsymbol{\xi}_k + \zeta_k \widehat{\nabla}_{\boldsymbol{\xi}} J_{\mathrm{P}}(\boldsymbol{\xi}_k)$. In particular, PGPE uses an estimator of $\nabla_{\boldsymbol{\xi}} J_{\mathrm{P}}(\boldsymbol{\xi})$ defined as:

$$\widehat{\nabla}_{\boldsymbol{\xi}} J_{\mathrm{P}}(\boldsymbol{\xi}) = \frac{1}{N} \sum_{i=0}^{N-1} \mathbf{g}_{\boldsymbol{\xi}}^{\mathrm{P}}(\boldsymbol{\theta}_i, \tau_i), \tag{5}$$

where N is the *batch size*, which in this context is the number of independent parameters-trajectories pairs $\{(\boldsymbol{\theta}_i, \tau_i)\}_{i=0}^{N-1}$, collected with hyperpolicy $\nu_{\boldsymbol{\xi}}$ ($\boldsymbol{\theta}_i \sim \nu_{\boldsymbol{\xi}}$ and $\tau_i \sim p_{\boldsymbol{\theta}_i}$). The single-parameter gradient estimator for the hyperpolicy is defined as follows:

$$\mathbf{g}_{\boldsymbol{\xi}}^{\mathrm{P}}(\boldsymbol{\theta}, \tau) \coloneqq \nabla \log \nu_{\boldsymbol{\xi}}(\boldsymbol{\theta}) R(\tau), \tag{6}$$

where $\tau \sim p_{\boldsymbol{\theta}}$.

A.2 Importance Sampling for Parameter-based Exploration

Consider to run a PGPE-like method for k iterations, collecting a set of hyperpolicy parameterizations $\{\boldsymbol{\xi}_i\}_{i=0}^{k-1}$ and, for each $\boldsymbol{\xi}_i$, sampling N policy parameterizations $\{\boldsymbol{\theta}_{i,j}\}_{j=0}^{N-1}$, i.e., $\forall j \in [\![0, N-1]\!]$: $\boldsymbol{\theta}_{i,j} \sim \nu_{\boldsymbol{\xi}_i}$. Consider each policy parameterization $\boldsymbol{\theta}_{i,j}$ to be used to sample a single trajectory $\tau_{i,j} \sim p_{\boldsymbol{\theta}_{i,j}}$. In this scenario, the data reused from previous iterates are the sampled policy parameterizations $\boldsymbol{\theta}_{i,j}$ collected under hyperpolicy $\boldsymbol{\xi}_i$ with their associated trajectories $\tau_{i,j} \sim p_{\boldsymbol{\theta}_{i,j}}$. The IW for incorporating this data into the gradient estimator is simply:

$$\frac{\nu_{\boldsymbol{\xi}_{k-1}}(\boldsymbol{\theta}_j) p_{\boldsymbol{\theta}_j}(\tau_j)}{\nu_{\boldsymbol{\xi}_i}(\boldsymbol{\theta}_j) p_{\boldsymbol{\theta}_j}(\tau_j)} = \frac{\nu_{\boldsymbol{\xi}_{k-1}}(\boldsymbol{\theta}_j)}{\nu_{\boldsymbol{\xi}_i}(\boldsymbol{\theta}_j)}. \tag{7}$$

That being said, the PB version of the PM estimator is defined as:

$$\widehat{\nabla}^{\mathrm{PM}} J_{\mathrm{P}}(\boldsymbol{\xi}_{k-1}) = \frac{1}{N} \sum_{i=0}^{k-1} \sum_{j=0}^{N-1} \frac{\alpha_{i,k} \nu_{\boldsymbol{\xi}_{k-1}}(\boldsymbol{\theta}_{i,j})}{(1-\lambda_{i,k}) \nu_{\boldsymbol{\xi}_i}(\boldsymbol{\theta}_{i,j}) + \lambda_{i,k} \nu_{\boldsymbol{\xi}_{k-1}}(\boldsymbol{\theta}_{i,j})} \mathbf{g}_{\boldsymbol{\xi}_{k-1}}^{\mathrm{P}}(\boldsymbol{\theta}_{i,j}, \tau_{i,j}).$$

A.3 Theoretical Guarantees of Parameter-based RPG

All the results presented in the main paper hold for the PB version of RPG, under assumptions that are the PB versions of Assumptions 4.1, 4.2, 4.3, and 4.4:

- Assumption 4.1 translates into requiring that there exist $G_{\mathrm{P}} < +\infty$ and $G_{2,\mathrm{P}} < +\infty$ such that

$$\sup_{\boldsymbol{\xi},\boldsymbol{\theta},\tau \in \Xi\times\Theta\times\mathcal{T}} \|\mathbf{g}^{\mathrm{P}}_{\boldsymbol{\xi}}(\boldsymbol{\theta},\tau)\|_2 \leqslant G_{\mathrm{P}} \quad \text{and} \quad \sup_{\boldsymbol{\xi},\boldsymbol{\theta},\tau \in \Xi\times\Theta\times\mathcal{T}} \|\nabla\mathbf{g}^{\mathrm{P}}_{\boldsymbol{\xi}}(\boldsymbol{\theta},\tau)\|_2 \leqslant G_{2,\mathrm{P}}. \tag{8}$$

- Assumption 4.2 translates into requiring that there exists $D_P \in \mathbb{R}_{\geqslant 1}$ such that

$$\sup_{\boldsymbol{\xi}_1,\boldsymbol{\xi}_2 \in \Xi} \chi^2(\nu_{\boldsymbol{\xi}_1}\|\nu_{\boldsymbol{\xi}_2}) \leqslant D_P - 1. \tag{9}$$

- Assumption 4.3 translates into requiring that there exists $L_{2,P} \in \mathbb{R}_{\geqslant 0}$ such that, for every $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2 \in \Xi$,

$$\|\nabla J_{\mathrm{P}}(\boldsymbol{\xi}_1) - \nabla J_{\mathrm{P}}(\boldsymbol{\xi}_2)\|_2 \leqslant L_{2,P}\|\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2\|_2. \tag{10}$$

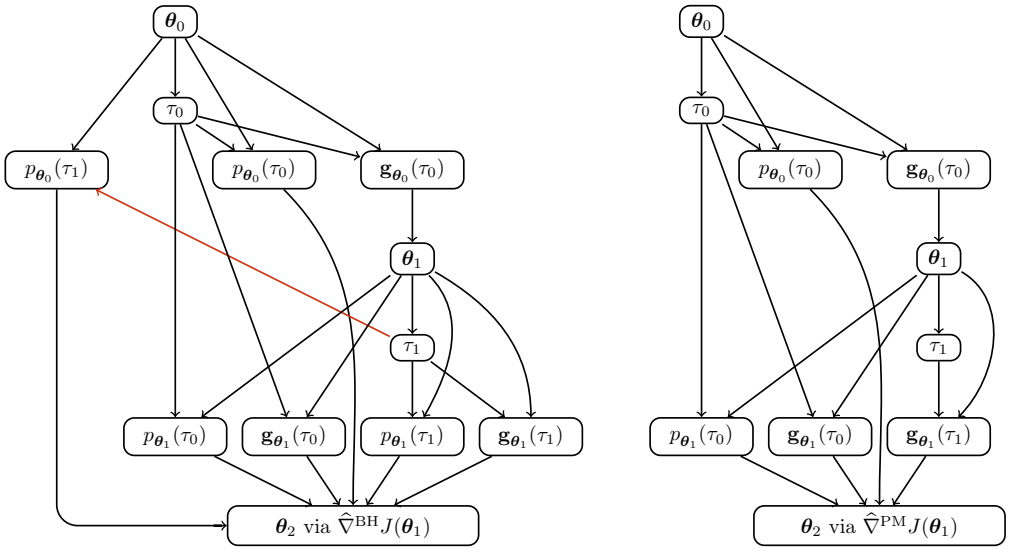
- Assumption 4.4 translates into requiring that there exist $L_{1,\nu}, L_{2,\nu} \in \mathbb{R}_{\geqslant 0}$ such that, for every $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2 \in \Xi$ and $\boldsymbol{\theta} \in \Theta$,

$$\|\log \nu_{\boldsymbol{\xi}_1}(\boldsymbol{\theta}) - \log \nu_{\boldsymbol{\xi}_2}(\boldsymbol{\theta})\|_2 \leqslant L_{1,\nu}\|\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2\|_2, \tag{11}$$

$$\|\nabla \log \nu_{\boldsymbol{\xi}_1}(\boldsymbol{\theta}) - \nabla \log \nu_{\boldsymbol{\xi}_2}(\boldsymbol{\theta})\|_2 \leqslant L_{2,\nu}\|\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2\|_2. \tag{12}$$

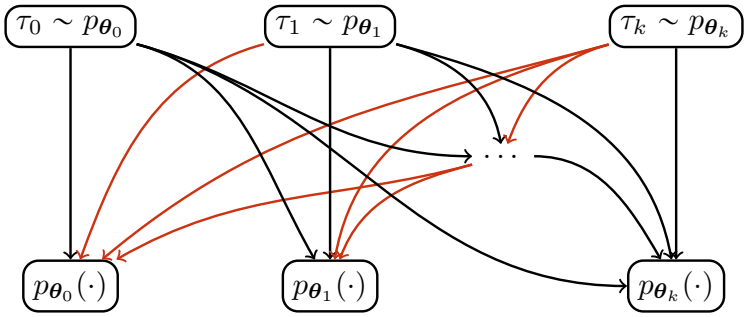
In particular, the χ^2 takes a simple form in the common case of Gaussian hyperpolicies, making it easier to ensure Assumption 4.2. (Metelli et al., 2020).

B On Temporal Dependencies in BH and PM Estimators

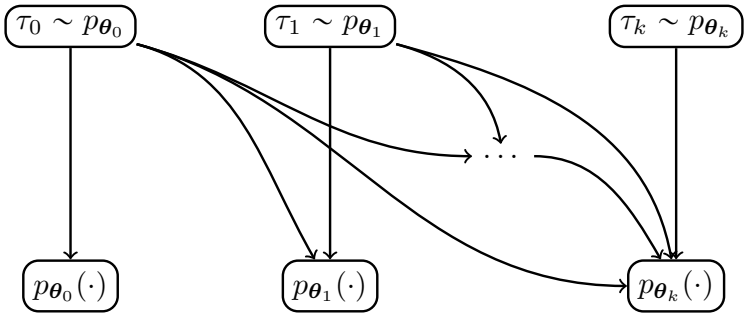


(a) BH Estimator (b) PM Estimator

Figure 3: Graphical models for the BH and PM estimators, considering $N = 1$ and $k = 2$ iterations. Nodes represent random variables, arrows represent causal relations. Elements in (—) represent anticausal dependencies.



(a) BH Estimator



(b) PM Estimator

Figure 4: Temporal dependencies among densities $p_{\boldsymbol{\theta}_i}$ ($i \in [\![0, k]\!]$) and sampled trajectories τ_j ($j \in [\![0, k]\!]$) of the BH and PM estimators with $N = 1$ and k iterations. Arrows represent required evaluations. Elements in (—) represent newer trajectories which have to be evaluated under older policies.

C Proofs of Section 4

Theorem 4.1 (Fixed Target PM Estimation Error Bound)**.** *Consider to run RPG for k iterations, collecting the parameterizations $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ with trajectories $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$. Let $\bar{\boldsymbol{\theta}} \in \Theta$ be chosen independently on $\{\boldsymbol{\theta}_i, \{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$. Under Assumptions 4.1 and 4.2, using the PM estimator with*

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log 6 + 4 \log \frac{1}{\delta}}{3D_i N k}} \quad \textit{and} \quad \alpha_{i,k} = \frac{D_i^{-1/2}}{\sum_{l=0}^{k-1} D_l^{-1/2}},$$

where $D_{k-1} \coloneqq 1$ and $D_i \coloneqq D$ for $i \in [\![0, k-2]\!]$, for every $\delta \in [0,1]$, with probability at least $1-\delta$, it holds that:

$$\left\| \widehat{\nabla}^{PM} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}}) \right\|_2 \leqslant 8G \sqrt{\frac{D d_\Theta \log 6 + D \log\left(\frac{1}{\delta}\right)}{Nk}}.$$

Proof. In order to study the concentration of $\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\|_2$, we will resort to Freedman's inequality, that we state below.

Theorem C.1 (Freedman's Inequality (Freedman, 1975))**.** *Let $(z_i)_{i=1}^m$ be a martingale difference sequence adapted to the filtration $(\mathcal{F}_{i-1})_{i=1}^m$ such that $|z_i| \leqslant M$ a.s. for every i and $\sum_{i=1}^m \mathbb{E}[z_i^2 | \mathcal{F}_{i-1}] \leqslant V$ (with M and V deterministic, possibly depending on m). Then, with probability $1-\delta$:*

$$\sum_{i=1}^m z_i \leqslant \sqrt{2V \log \frac{1}{\delta}} + \frac{2}{3} M \log \frac{1}{\delta}. \tag{13}$$

Furthermore, we focus on the inner product between the gradient estimator and a fixed unit vector w (i.e., $\|w\|_2 = 1$). For $i \in [\![0, k-1]\!]$ and $j \in [\![0, N-1]\!]$, let us define:

$$x_{i,j} = \frac{\alpha_{i,k}}{N} \frac{w^\top g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{(1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} + \lambda_{i,k}}, \quad z_{i,j} = x_{i,j} - \mathbb{E}[x_{i,j} | \mathcal{F}_{i-1}], \tag{14}$$

where $\mathcal{F}_{i-1} = \sigma(\boldsymbol{\theta}_0, \{\tau_{0,j}\}_{j=0}^{N-1}, \ldots, \boldsymbol{\theta}_{k-2}, \{\tau_{k-2,j}\}_{j=0}^{N-1}, \boldsymbol{\theta}_{k-1})$ is the filtration. Notice that the filtration depends on i only, since, within the batch, the trajectories are independent. Furthermore, we have that $\mathbb{E}[x_{i,j} | \mathcal{F}_{i-1}] = \mathbb{E}[x_{i,j'} | \mathcal{F}_{i-1}]$ for every $j, j' \in [\![0, N-1]\!]$. Given this, we have:

$$w^\top \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) = \sum_{i=0}^{k-1} \sum_{j=0}^{N-1} x_{i,j}. \tag{15}$$

First of all, we observe the boundedness for every i, j:

$$|x_{i,j}| \leqslant \frac{\alpha_{i,k} G(w)}{\lambda_{i,k} N}, \qquad |z_{i,j}| \leqslant \frac{2\alpha_{i,k} G(w)}{\lambda_{i,k} N}, \tag{16}$$

a.s., being $G(w) \coloneqq \sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} w^\top g_{\boldsymbol{\theta}}(\tau)$. We now prove that $((z_{i,j})_{j=0}^{N-1})_{i=0}^{k-1}$ is a martingale difference sequence. Indeed, for $i \in [\![0, k-1]\!]$ and $j \in [\![0, N-1]\!]$, we have:

$$\mathbb{E}[|z_{i,j}|] \leqslant \frac{2\alpha_{i,k} G(w)}{N \lambda_{i,k}} < +\infty, \tag{17}$$

$$\mathbb{E}[z_{i,j} | \mathcal{F}_{i-1}] = \mathbb{E}[x_{i,j} - \mathbb{E}[x_{i,j} | \mathcal{F}_{i-1}] | \mathcal{F}_{i-1}] = 0, \tag{18}$$

a.s.. Let us now compute the second moment:

$$\mathbb{E}[z_{i,j}^2 | \mathcal{F}_{i-1}] \leqslant \mathbb{E}[x_{i,j}^2 | \mathcal{F}_{i-1}] \leqslant \frac{\alpha_{i,k}^2 G(w)^2 D_i}{N^2}, \tag{19}$$

where

$$D_i = \begin{cases} 1 & \text{if } i = k-1 \\ D & \text{otherwise} \end{cases} \tag{20}$$

since, conditioned to $\mathcal{F}_{i-1}$, this is the standard power mean estimator, whose variance has been established in (Lemma 5.1, Metelli et al., 2021). For what concerns the bias, let us define:

$$y_{i,j} = x_{i,j}|_{\lambda_{i,k}=0} = \frac{\alpha_{i,k}}{N} \frac{w^\top g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{\frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}}. \tag{21}$$

Note that: $\mathbb{E}[y_{i,j}|\mathcal{F}_{i-1}] = w^\top \nabla J(\bar{\boldsymbol{\theta}})$ for every i, j. Thus:

$$|\,\mathbb{E}[x_{i,j}|\mathcal{F}_{i-1}] - w^\top \nabla J(\bar{\boldsymbol{\theta}})| = |\,\mathbb{E}[x_{i,j}|\mathcal{F}_{i-1}] - \mathbb{E}[y_{i,j}|\mathcal{F}_{i-1}]| \leqslant \frac{G(w)\alpha_{i,k}\lambda_{i,k}D_i}{N}, \tag{22}$$

since, when conditioning to $\mathcal{F}_{i-1}$, we are evaluating the bias of a PM estimator, whose has been established again in (Lemma 5.1, Metelli et al., 2021). In order to apply Freedman's inequality, we have to guarantee that the bounds on the variance and maximum value of the martingale difference sequence are deterministic. Thus, we can choose $\alpha_{i,k}$ and $\lambda_{i,k}$ based on the index i (possibly j), but not on the history. We choose:

$$\lambda_{i,k} = \sqrt{\frac{4\log\frac{1}{\delta}}{3D_i N k}}, \qquad \alpha_{i,k} = \frac{D_i^{-1/2}}{\sum_{l=0}^{k-1} D_l^{-1/2}}. \tag{23}$$

Notice that this property ensures that $\frac{\alpha_{i,k}}{\lambda_{i,k}}$ is a constant independent on i. Thus, w.p. $1-\delta$, we have:

$$w^\top(\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})) \tag{24}$$

$$= w^\top \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \sum_{i=0}^{k-1}\sum_{j=0}^{N-1} \mathbb{E}[x_{i,j}|\mathcal{F}_{i-1}] + \sum_{i=0}^{k-1}\sum_{j=0}^{N-1} \mathbb{E}[x_{i,j}|\mathcal{F}_{i-1}] - w^\top \nabla J(\bar{\boldsymbol{\theta}}) \tag{25}$$

$$\leqslant \sum_{i=0}^{k-1}\sum_{j=0}^{N-1} z_{i,j} + \sum_{i=0}^{k-1}\sum_{j=0}^{N-1} |\,\mathbb{E}[x_{i,j}|\mathcal{F}_{i-1}] - w^\top \nabla J(\bar{\boldsymbol{\theta}})| \tag{26}$$

$$\leqslant G(w)\sqrt{\frac{2}{N}\sum_{i=0}^{k-1}\alpha_{i,k}^2 D_i \log\frac{1}{\delta}} + \frac{4G(w)}{3Nk}\sum_{i=0}^{k-1}\frac{\alpha_{i,k}}{\lambda_{i,k}}\log\frac{1}{\delta} + G(w)\sum_{i=0}^{k-1}\alpha_{i,k}\lambda_{i,k}D_i. \tag{27}$$

By replacing our choices of $\lambda_{i,k}$ and $\alpha_{i,k}$:

$$\frac{G(w)}{\sum_{i=0}^{k-1} D_i^{-1/2}}\sqrt{\frac{k}{N}\log\frac{1}{\delta}}\underbrace{(\sqrt{2}+\sqrt{4/3}+\sqrt{4/3})}_{\leqslant 4} \leqslant 4G(w)\sqrt{\frac{D}{Nk}\log\frac{1}{\delta}}, \tag{28}$$

having bounded all $D_i \leqslant D$ in the inequality for $i \in [\![0, k-1]\!]$. To bound the norm, we follow the standard approach, defining $\mathcal{C}_\eta$ as an η-cover (with $\eta < 1$) of the unit ball (i.e., $\sup_{w:\|w\|_2\leqslant 1}\inf_{w'\in\mathcal{C}_\eta}\|w-w'\|_2 \leqslant \eta$) having cardinality $|C_\eta| \leqslant (3/\eta)^{d_\Theta}$, and observing that:

$$\left\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2 \tag{29}$$

$$= \sup_{w\,:\,\|w\|_2=1} w^\top(\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})) \tag{30}$$

$$\leqslant \sup_{w\,:\,\|w\|_2=1}\inf_{w'\in\mathcal{C}_\eta}\left\{(w')^\top(\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})) + (w-w')^\top(\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}}))\right\} \tag{31}$$

$$\leqslant \sup_{w\in\mathcal{C}_\eta} w^\top(\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})) + \eta\left\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2 \tag{32}$$

$$= (1-\eta)^{-1}\sup_{w\in\mathcal{C}_\eta} w^\top(\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})). \tag{33}$$

With a union bound over the points of the cover, we have w.p. $1-\delta$:

$$\left\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2 \leqslant (1-\eta)^{-1} 4G\sqrt{\frac{D}{Nk}\log\frac{|\mathcal{C}_\eta|}{\delta}} \tag{34}$$

$$\leqslant (1-\eta)^{-1} 4G\sqrt{\frac{Dd_\Theta\log\left(\frac{3}{\eta}\right) + D\log\left(\frac{1}{\delta}\right)}{Nk}} \tag{35}$$

where $G = \sup_{w \in \mathcal{C}_\eta} G(w) = \sup_{w \in \mathcal{C}_\eta} \sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} \omega^\top \mathbf{g}_{\boldsymbol{\theta}}(\tau) \leqslant \sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} \|g_{\boldsymbol{\theta}}(\tau)\|_2 = G$, as defined above. We choose $\eta = 1/2$, obtaining:

$$\left\| \widehat{\nabla}^{\mathrm{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}}) \right\|_2 \leqslant 8G \sqrt{\frac{D d_\Theta \log 6 + D \log\left(\frac{1}{\delta}\right)}{Nk}}. \tag{36}$$

We conclude by noticing that, after the union bound, the coefficients $\lambda_{i,k}$ for every $i \in [\![0, k-1]\!]$ become:

$$\lambda_{i,k} = \sqrt{\frac{4 \log\left(\frac{|\mathcal{C}_{1/2}|}{\delta}\right)}{3 D_i N k}} \tag{37}$$

$$= \sqrt{\frac{4 d_\Theta \log 6 + 4 \log \frac{1}{\delta}}{3 D_i N k}}. \tag{38}$$

□

Lemma C.2 (Characterization of G and G_2)**.** *Under Assumption 4.4, the following hold:*

$$\sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} \left\| \boldsymbol{g}^R_{\boldsymbol{\theta}}(\tau) \right\|_2 \leqslant G_R \coloneqq \frac{T(1-\gamma^T)}{1-\gamma} R_{\max} L_{1,\pi},$$

$$\sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} \left\| \nabla \boldsymbol{g}^R_{\boldsymbol{\theta}}(\tau) \right\|_2 \leqslant G_{2,R} \coloneqq \frac{T(1-\gamma^T)}{1-\gamma} R_{\max} L_{2,\pi},$$

$$\sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} \left\| \boldsymbol{g}^G_{\boldsymbol{\theta}}(\tau) \right\|_2 \leqslant G_G \coloneqq \frac{1-\gamma^T}{(1-\gamma)^2} R_{\max} L_{1,\pi},$$

$$\sup_{\boldsymbol{\theta}, \tau \in \Theta \times \mathcal{T}} \left\| \nabla \boldsymbol{g}^G_{\boldsymbol{\theta}}(\tau) \right\|_2 \leqslant G_{2,G} \coloneqq \frac{1-\gamma^T}{(1-\gamma)^2} R_{\max} L_{2,\pi}.$$

Proof. These results simply come from the explicit forms of $\mathbf{g}^{\mathrm{R}}_{\boldsymbol{\theta}}(\tau)$ and $\mathbf{g}^{\mathrm{G}}_{\boldsymbol{\theta}}(\tau)$, then applying Assumption 4.4. Similar results are presented in (Papini et al., 2022).

For REINFORCE, we have:

$$\left\| \mathbf{g}^{\mathrm{R}}_{\boldsymbol{\theta}}(\tau) \right\|_2 = \left\| \sum_{t=0}^{T-1} \nabla \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,t} | \mathbf{s}_{\tau,t}) R(\tau) \right\|_2 \leqslant \frac{T(1-\gamma^T)}{1-\gamma} R_{\max} L_{1,\pi}, \tag{39}$$

and

$$\left\| \nabla \mathbf{g}^{\mathrm{R}}_{\boldsymbol{\theta}}(\tau) \right\|_2 = \left\| \nabla \sum_{t=0}^{T-1} \nabla \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,t} | \mathbf{s}_{\tau,t}) R(\tau) \right\|_2 \leqslant \frac{T(1-\gamma^T)}{1-\gamma} R_{\max} L_{2,\pi}. \tag{40}$$

Similarly, for GPOMDP, the following holds:

$$\left\| \mathbf{g}^{\mathrm{G}}_{\boldsymbol{\theta}}(\tau) \right\|_2 = \left\| \sum_{t=0}^{T-1} \left(\sum_{l=0}^{t} \nabla \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,l} | \mathbf{s}_{\tau,l}) \right) \gamma^t r(\mathbf{s}_{\tau,t}, \mathbf{a}_{\tau,t}) \right\|_2 \leqslant \frac{1-\gamma^T}{(1-\gamma)^2} R_{\max} L_{1,\pi}, \tag{41}$$

and

$$\left\| \nabla \mathbf{g}^{\mathrm{G}}_{\boldsymbol{\theta}}(\tau) \right\|_2 = \left\| \nabla \sum_{t=0}^{T-1} \left(\sum_{l=0}^{t} \nabla \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{\tau,l} | \mathbf{s}_{\tau,l}) \right) \gamma^t r(\mathbf{s}_{\tau,t}, \mathbf{a}_{\tau,t}) \right\|_2 \leqslant \frac{1-\gamma^T}{(1-\gamma)^2} R_{\max} L_{2,\pi}. \tag{42}$$

□

Lemma C.3. *Suppose to employ the PM estimator with policy parameterizations $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ and related trajectories $\{\tau_{i,j}\}_{j=0}^{N-1}$, i.e., for any $i \in [\![0, k-1]\!]$ and $j \in [\![0, N-1]\!]$, $\tau_{i,j} \sim p_{\boldsymbol{\theta}_i}$. Under Assumptions 4.1, 4.2, 4.3, and 4.4, for every pair of parameterizations $\bar{\boldsymbol{\theta}}_1, \bar{\boldsymbol{\theta}}_2 \in \Theta$, using the choices of $\alpha_{i,k}$ and $\lambda_{i,k}$ from Theorem 4.1, the following holds:*

$$\left\| \left(\widehat{\nabla}^{PM} J(\bar{\boldsymbol{\theta}}_1) - \nabla J(\bar{\boldsymbol{\theta}}_1) \right) - \left(\widehat{\nabla}^{PM} J(\bar{\boldsymbol{\theta}}_2) - \nabla J(\bar{\boldsymbol{\theta}}_2) \right) \right\|_2 \leqslant L_{PM} \left\| \bar{\boldsymbol{\theta}}_1 - \bar{\boldsymbol{\theta}}_2 \right\|_2,$$

where

$$L_{PM} \coloneqq L_{2,J} + (GTL_{1,\pi} + G_2) \sqrt{\frac{3}{4} DNk}.$$

Proof. We start the proof with the following derivation:

$$\left\| (\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_1) - \nabla J(\bar{\boldsymbol{\theta}}_1)) - (\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_2) - \nabla J(\bar{\boldsymbol{\theta}}_2)) \right\|_2 \tag{43}$$

$$\leqslant \left\| \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_1) - \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_2) \right\|_2 + \left\| \nabla J(\bar{\boldsymbol{\theta}}_1) - \nabla J(\bar{\boldsymbol{\theta}}_2) \right\|_2 \tag{44}$$

$$\leqslant \left\| \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_1) - \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_2) \right\|_2 + L_{2,J} \left\| \bar{\boldsymbol{\theta}}_1 - \bar{\boldsymbol{\theta}}_2 \right\|_2, \tag{45}$$

where we used the triangular inequality and we exploited Assumption 4.3.

Now, in order to deal with $\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_1) - \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}_2)\|_2$, we can equivalently bound $\|\nabla_{\bar{\boldsymbol{\theta}}} \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}})\|_2$, for any $\bar{\boldsymbol{\theta}} \in \Theta$.

$$\left\| \nabla_{\bar{\boldsymbol{\theta}}} \widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) \right\|_2 \tag{46}$$

$$= \left\| \nabla_{\bar{\boldsymbol{\theta}}} \left(\frac{1}{N} \sum_{i=0}^{k-1} \sum_{j=0}^{N-1} \frac{\alpha_{i,k}}{(1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} + \lambda_{i,k}} g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j}) \right) \right\|_2 \tag{47}$$

$$\leqslant \left\| \frac{1}{N} \sum_{i=0}^{k-1} \sum_{j=0}^{N-1} \left(\frac{\alpha_{i,k}(1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})^2} \nabla_{\bar{\boldsymbol{\theta}}} p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j}) g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{\left((1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} + \lambda_{i,k} \right)^2} + \frac{\alpha_{i,k} \nabla_{\bar{\boldsymbol{\theta}}} g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{(1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} + \lambda_{i,k}} \right) \right\|_2, \tag{48}$$

obtained by simply making the form of the PM estimator explicit and by computing the gradient w.r.t. $\bar{\boldsymbol{\theta}}$.

Before carrying on with the derivation, note that

$$\frac{\nabla_{\bar{\boldsymbol{\theta}}} p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} = \nabla_{\bar{\boldsymbol{\theta}}} \log p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j}), \tag{49}$$

that

$$\frac{1}{(1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} + \lambda_{i,k}} (1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} \leqslant 1, \tag{50}$$

and that

$$\frac{\alpha_{i,k}}{(1-\lambda_{i,k}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})} + \lambda_{i,k}} \leqslant \frac{\alpha_{i,k}}{\lambda_{i,k}}. \tag{51}$$

That being said, we have what follows:

$$\left\|\nabla_{\bar{\boldsymbol{\theta}}}\widehat{\nabla}^{\mathrm{PM}}J(\bar{\boldsymbol{\theta}})\right\|_2 \tag{52}$$

$$\leqslant \left\|\frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\left(\frac{\alpha_{i,k}(1-\lambda_{i,k})\frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})^2}\nabla_{\bar{\boldsymbol{\theta}}}p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{\left((1-\lambda_{i,k})\frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}+\lambda_{i,k}\right)^2}+\frac{\alpha_{i,k}\nabla_{\bar{\boldsymbol{\theta}}}g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}{(1-\lambda_{i,k})\frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}+\lambda_{i,k}}\right)\right\|_2 \tag{53}$$

$$\leqslant \frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\left(\frac{\alpha_{i,k}\left\|\nabla_{\bar{\boldsymbol{\theta}}}\log p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2\left\|g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2}{(1-\lambda_{i,k})\frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}+\lambda_{i,k}}+\frac{\alpha_{i,k}\left\|\nabla_{\bar{\boldsymbol{\theta}}}g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2}{(1-\lambda_{i,k})\frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})}+\lambda_{i,k}}\right) \tag{54}$$

$$\leqslant \underbrace{\frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{\alpha_{i,k}}{\lambda_{i,k}}\left\|g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2\left\|\nabla_{\bar{\boldsymbol{\theta}}}\log p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2}_{\mathsf{A}}+\underbrace{\frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{\alpha_{i,k}}{\lambda_{i,k}}\left\|\nabla_{\bar{\boldsymbol{\theta}}}g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2}_{\mathsf{B}}, \tag{55}$$

where we used inequalities introduced above.

Before continuing with the derivation, we exploit the choices for the $\alpha_{i,k}$ and $\lambda_{i,k}$ terms:

$$\lambda_{i,k}=\sqrt{\frac{4d_\Theta\log 6+4\log\frac{1}{\delta}}{3D_iNk}}\quad\text{and}\quad\alpha_{i,k}=\frac{D_i^{-1/2}}{\sum_{l=0}^{k-1}D_l^{-1/2}}, \tag{56}$$

being $1-\delta$ the probability with which the bound of Theorem 4.1 holds and

$$D_i=\begin{cases}1 & \text{If } i=k-1\\ D & \text{otherwise}\end{cases}. \tag{57}$$

This choice for $\alpha_{i,k}$ and $\lambda_{i,k}$ leads to the following constant ratios for every $i\in[\![0,k-1]\!]$:

$$\frac{\alpha_{i,k}}{\lambda_{i,k}}\leqslant\sqrt{\frac{3ND}{4k\left(d_\Theta\log 6+\log\frac{1}{\delta}\right)}}\leqslant\sqrt{\frac{3ND}{4k}}. \tag{58}$$

Now, exploiting this bound on $\alpha_{i,k}/\lambda_{i,k}$, we focus on the first term A:

$$\mathsf{A}=\frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{\alpha_{i,k}}{\lambda_{i,k}}\left\|g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2\left\|\nabla_{\bar{\boldsymbol{\theta}}}\log p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2 \tag{59}$$

$$\leqslant G\sqrt{\frac{3D}{4Nk}}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\left\|\nabla_{\bar{\boldsymbol{\theta}}}\log p_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2 \tag{60}$$

$$\leqslant G\sqrt{\frac{3D}{4Nk}}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\sum_{l=0}^{T-1}\left\|\nabla_{\bar{\boldsymbol{\theta}}}\log\pi_{\bar{\boldsymbol{\theta}}}(\mathbf{a}_{\tau_{i,j},l}|\mathbf{s}_{\tau_{i,j},l})\right\|_2 \tag{61}$$

$$\leqslant GTL_{1,\pi}\sqrt{\frac{3}{4}DNk}, \tag{62}$$

where we exploited Assumptions 4.1 and 4.4.

To conclude the derivation, we can now focus on term B. We will exploit the bound on $\alpha_{i,k}/\lambda_{i,k}$ and Assumption 4.1. The following holds:

$$\mathsf{B}=\frac{1}{N}\sum_{i=0}^{k-1}\sum_{j=0}^{N-1}\frac{\alpha_{i,k}}{\lambda_{i,k}}\left\|\nabla_{\bar{\boldsymbol{\theta}}}g_{\bar{\boldsymbol{\theta}}}(\tau_{i,j})\right\|_2 \tag{63}$$

$$\leqslant G_2\sqrt{\frac{3}{4}DNk}. \tag{64}$$

All in all, we have:

$$\left\|(\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_1) - \nabla J(\boldsymbol{\theta}_1)) - (\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_2) - \nabla J(\boldsymbol{\theta}_2))\right\|_2 \tag{65}$$

$$\leqslant \left\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_1) - \widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_2)\right\|_2 + L_{2,J}\left\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\right\|_2 \tag{66}$$

$$\leqslant L_{\text{PM}}\left\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\right\|_2, \tag{67}$$

where

$$L_{\text{PM}} \coloneqq L_{2,J} + (GTL_{1,\pi} + G_2)\sqrt{\frac{3}{4}DNk}. \tag{68}$$

Before concluding the proof, we let the reader note that when $T = +\infty$ (and $\gamma < 1$), we identify the length of a trajectory with the effective horizon $T \approx \widetilde{\mathcal{O}}(1/(1-\gamma))$. This approximation only affects logarithmic terms in the sample complexity (Yuan et al., 2022). □

Theorem 4.2. *Consider to run RPG for k iterations with a constant step size ζ, collecting the parameterizations $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ with trajectories $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$. Under Assumptions 4.1, 4.2, 4.3, and 4.4, select the $\alpha_{i,k}$ terms as in Theorem 4.1 and, for every $\delta \in [0,1]$, the $\lambda_{i,k}$ terms as:*

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left((18\sqrt{3}L_{2,J} + 27L_1\sqrt{DNk})\frac{\zeta Nk^2}{16\delta\sqrt{d_\Theta}}\right) + 4\log\frac{1}{\delta}}{3D_iNk}},$$

where $L_1 \coloneqq GTL_{1,\pi} + G_2$. With probability at least $1-\delta$, it holds:

$$\left\|\widehat{\nabla}^{PM}J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1})\right\|_2 \leqslant 16G\sqrt{\frac{Dd_\Theta}{Nk}\log\left(6\zeta\left(L_{2,J} + L_1\sqrt{\frac{3}{4}DNk}\right)\frac{Nk^2}{\delta\sqrt{d_\Theta}}\right)}.$$

Proof. Consider to be at iteration $k \in \mathbb{N}$ of RPG. Additionally, consider the k^{th} parameterization $\boldsymbol{\theta}_{k-1}$ to belong to a d_Θ-dimensional ball $\mathcal{B}_\rho^{d_\Theta}$ with radius $\rho \in \mathbb{R}_{>0}$, i.e., $\boldsymbol{\theta}_{k-1} \in \mathcal{B}_\rho^{d_\Theta}$.

Covering of LC Functions. Let $\mathcal{C}_{\eta_k}$ be a k-dependent η_k-cover (with $\eta_k \leqslant \rho$) of $\mathcal{B}_\rho^{d_\Theta}$. Thus, for every parameterization $\boldsymbol{\theta} \in \mathcal{B}_\rho^{d_\Theta}$ there exists $\boldsymbol{c} \in \mathcal{C}_{\eta_k}$ such that $\|\boldsymbol{\theta} - \boldsymbol{c}\|_2 \leqslant \eta_k$. The cardinality of the cover set is finite and bounded as:

$$|\mathcal{C}_{\eta_k}| \leqslant \left(1 + \frac{2\rho}{\eta_k}\right)^{d_\Theta} \leqslant \left(\frac{3\rho}{\eta_k}\right)^{d_\Theta}. \tag{69}$$

Now, let $\mathbf{x} \in \mathcal{B}_\rho^{d_\Theta}$ and let $\mathbf{x}_c \in \mathcal{C}_{\eta_k}$ such that $\|\mathbf{x} - \mathbf{x}_c\|_2 \leqslant \eta_k$. For any L-LC function $f : \mathcal{B}_\rho^{d_\Theta} \to \mathbb{R}^{d_\Theta}$ the following holds:

$$\|f(\mathbf{x})\|_2 \leqslant \|f(\mathbf{x}) - f(\mathbf{x}_c)\|_2 + \|f(\mathbf{x}_c)\|_2 \leqslant L\eta_k + \|f(\mathbf{x}_c)\|_2, \tag{70}$$

which follows by the triangular inequality and the cover definition.

Covering over $\boldsymbol{\theta}$ of the PM Estimation Error. In what follows, we need to apply both Theorem 4.1 and Lemma C.3, for which we need to select the $\alpha_{i,k}$ and $\lambda_{i,k}$ terms as:

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta\log 6 + 4\log\frac{1}{\delta}}{3D_iNk}} \quad \text{and} \quad \alpha_{i,k} = \frac{D_i^{-1/2}}{\sum_{l=0}^{k-1} D_l^{-1/2}}, \tag{71}$$

being $1-\delta$ the probability with which the bound of Theorem 4.1 holds and

$$D_i = \begin{cases} 1 & \text{If } i = k-1 \\ D & \text{otherwise} \end{cases}. \tag{72}$$

We can apply Lemma C.3 which states that the function $\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}) - \nabla J(\boldsymbol{\theta})$ is L_{PM}-LC. Thus, the following holds:

$$\left\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1})\right\|_2 \leqslant \sup_{\bar{\boldsymbol{\theta}} \in \mathcal{B}_\rho^{d_\Theta}} \left\|\widehat{\nabla}^{\text{PM}}J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2 \tag{73}$$

$$\leqslant \eta_k L_{\text{PM}} + \max_{\bar{\boldsymbol{\theta}} \in \mathcal{C}_{\eta_k}} \left\|\widehat{\nabla}^{\text{PM}}J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2, \tag{74}$$

where the first inequality consists of a uniform bound over the parameterizations $\bar{\boldsymbol{\theta}} \in \mathcal{B}_{\rho}^{d_\Theta}$.

Now that $\bar{\boldsymbol{\theta}}$ is independent on the parameterization history $\{\boldsymbol{\theta}_i\}_{i=0}^{k-2}$, we can apply Theorem 4.1, stating that, w.p. $1-\delta'$, the following holds:

$$\left\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2 \leqslant 8G\sqrt{\frac{Dd_\Theta \log 6 + D\log\left(\frac{1}{\delta'}\right)}{Nk}}. \tag{75}$$

Performing a union bound over all the parameterization $\bar{\boldsymbol{\theta}} \in \mathcal{C}_{\eta_k}$, w.p. $1-\delta$, the following holds:

$$\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1})\right\|_2 \leqslant \eta_k L_{\text{PM}} + \max_{\bar{\boldsymbol{\theta}}\in\mathcal{C}_{\eta_k}} \left\|\widehat{\nabla}^{\text{PM}} J(\bar{\boldsymbol{\theta}}) - \nabla J(\bar{\boldsymbol{\theta}})\right\|_2 \tag{76}$$

$$\leqslant \eta_k L_{\text{PM}} + 8G\sqrt{\frac{Dd_\Theta \log 6 + D\log\left(\frac{|\mathcal{C}_{\eta_k}|}{\delta}\right)}{Nk}} \tag{77}$$

$$\leqslant \eta_k L_{\text{PM}} + 8G\sqrt{\frac{Dd_\Theta \log\left(\frac{18\rho}{\eta_k}\right) + D\log\left(\frac{1}{\delta}\right)}{Nk}} \tag{78}$$

$$\leqslant \eta_k L_{\text{PM}} + 8G\sqrt{\frac{Dd_\Theta}{Nk}\log\left(\frac{18\rho}{\eta_k\delta}\right)}. \tag{79}$$

Notice that, having performed a union bound over the cover set $\mathcal{C}_{\eta_k}$, the coefficient $\lambda_{i,k}$ has to be modified accordingly as:

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log 6 + 4\log\frac{|\mathcal{C}_{\eta_k}|}{\delta}}{3D_i Nk}} \tag{80}$$

$$= \sqrt{\frac{4d_\Theta \log\left(\frac{18\rho}{\eta_k}\right) + 4\log\frac{1}{\delta}}{3D_i Nk}}. \tag{81}$$

The last thing to do to conclude the application of the covering argument is to select the value for η_k. In particular, we select η_k as:

$$\eta_k = \frac{8G}{L_{\text{PM}}}\sqrt{\frac{Dd_\Theta}{Nk}}. \tag{82}$$

Notice that we should enforce $\eta_k \leqslant \rho$. However, the ball radius ρ will be selected later in this proof to be larger than η_k.

Substituting this value for η_k, we obtain what follows:

$$\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1})\right\|_2 \leqslant \eta_k L_{\text{PM}} + 8G\sqrt{\frac{Dd_\Theta}{Nk}\log\left(\frac{18\rho}{\eta_k\delta}\right)} \tag{83}$$

$$= 8G\sqrt{\frac{Dd_\Theta}{Nk}} + 8G\sqrt{\frac{Dd_\Theta}{Nk}\log\left(\frac{18\rho L_{\text{PM}}\sqrt{Nk}}{8G\delta\sqrt{Dd_\Theta}}\right)}, \tag{84}$$

which holds w.p. $1-\delta$.

Radius Selection. The next step consists in selecting an appropriate value for the ball radius ρ. Given that we are at iteration k of RPG, we have to select ρ in order to ensure that $\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_0\|_2 \leqslant \rho$. In order to do this, let us consider the maximum displacement between two subsequent parameterizations

$\boldsymbol{\theta}_z$ and $\boldsymbol{\theta}_{z-1}$ ($z \in [\![k-1]\!]$):

$$\|\boldsymbol{\theta}_{z+1} - \boldsymbol{\theta}_z\|_2 = \zeta \left\| \widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_z) \right\|_2 \tag{85}$$

$$= \zeta \left\| \frac{1}{N} \sum_{i=0}^{z} \sum_{j=0}^{N-1} \frac{\alpha_{i,z}}{(1-\lambda_{i,z}) \frac{p_{\boldsymbol{\theta}_i}(\tau_{i,j})}{p_{\boldsymbol{\theta}_z}(\tau_{i,j})} + \lambda_{i,z}} \mathbf{g}_{\boldsymbol{\theta}_z}(\tau_{i,j}) \right\|_2 \tag{86}$$

$$\leqslant \zeta G \sqrt{\frac{3DNz}{4d_\Theta \log\left(\frac{18\rho}{\eta_k}\right) + 4\log\frac{1}{\delta}}} \tag{87}$$

$$\leqslant \zeta G \sqrt{\frac{3DNk}{4d_\Theta \log\left(\frac{18\rho}{\eta_k}\right) + 4\log\frac{1}{\delta}}} \tag{88}$$

$$\leqslant \zeta G \sqrt{\frac{3}{4} DNk}, \tag{89}$$

given the selection of the terms $\alpha_{i,k}$ and $\lambda_{i,k}$. Importantly, in the last inequality we recover the upper bound on the ratios $\alpha_{i,k}/\lambda_{i,k}$ shown in the derivation of Lemma C.3, i.e., the value we would obtain without performing the union bound on the cover set $\mathcal{C}_{\eta_k}$, which just enlarge the denominator. We let the reader note that this passage is crucial, since without it we would have a circular dependency for the radius ρ selection. We further comment that, given this argument, the term L_{PM} preserves the same expression reported in Lemma C.3. We thus select the ball radius ρ as the maximum value for $\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_0\|$:

$$\|\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_0\|_2 \leqslant \sum_{z=0}^{k-2} \|\boldsymbol{\theta}_{z+1} - \boldsymbol{\theta}_z\|_2 \leqslant \zeta G k \sqrt{\frac{3}{4} DNk} =: \rho. \tag{90}$$

We can finally plug the choice of ρ into Line (84), obtaining the following:

$$\left\| \widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1}) \right\|_2 \leqslant 8G\sqrt{\frac{Dd_\Theta}{Nk}} + 8G\sqrt{\frac{Dd_\Theta}{Nk} \log\left(\frac{18\rho L_{\text{PM}}\sqrt{Nk}}{8G\delta\sqrt{Dd_\Theta}}\right)} \tag{91}$$

$$\leqslant 8G\sqrt{\frac{Dd_\Theta}{Nk}} + 8G\sqrt{\frac{Dd_\Theta}{Nk} \log\left(\frac{2\zeta L_{\text{PM}} Nk^2}{\delta\sqrt{d_\Theta}}\right)} \tag{92}$$

$$= 8G\sqrt{\frac{Dd_\Theta}{Nk}} \left(1 + \sqrt{\log\left(\frac{2\zeta L_{\text{PM}} Nk^2}{\delta\sqrt{d_\Theta}}\right)}\right) \tag{93}$$

$$\leqslant 16G\sqrt{\frac{Dd_\Theta}{Nk}} \sqrt{1 + \log\left(\frac{2\zeta L_{\text{PM}} Nk^2}{\delta\sqrt{d_\Theta}}\right)} \tag{94}$$

$$\leqslant 16G\sqrt{\frac{Dd_\Theta}{Nk} \log\left(\frac{2e\zeta L_{\text{PM}} Nk^2}{\delta\sqrt{d_\Theta}}\right)} \tag{95}$$

$$\leqslant 16G\sqrt{\frac{Dd_\Theta}{Nk} \log\left(\frac{6\zeta L_{\text{PM}} Nk^2}{\delta\sqrt{d_\Theta}}\right)}, \tag{96}$$

which holds w.p. $1-\delta$. We let the reader note that in the last inequalities we exploited the fact that $1 + \sqrt{\log(x)} \leqslant 2\sqrt{\log(ex)}$, for any $x \in \mathbb{R}_{\geqslant 1}$.

To conclude the proof, we have to substitute the value of L_{PM}, obtaining the following result:

$$\left\| \widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_{k-1}) - \nabla J(\boldsymbol{\theta}_{k-1}) \right\|_2 \tag{97}$$

$$\leqslant 16G\sqrt{\frac{Dd_\Theta}{Nk} \log\left(\frac{6\zeta\left(L_{2,J} + (GTL_{1,\pi} + G_2)\sqrt{\frac{3}{4}DNk}\right) Nk^2}{\delta\sqrt{d_\Theta}}\right)}. \tag{98}$$

Notice that the final value for the $\lambda_{i,k}$ terms, substituting the found values for ρ, η_k, and L_{PM} is:

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left(\frac{18\rho}{\eta_k}\right) + 4\log\frac{1}{\delta}}{3D_i Nk}} \tag{99}$$

$$= \sqrt{\frac{4d_\Theta \log\left(\frac{18L_{\text{PM}}\sqrt{Nk}\rho}{8G\sqrt{Dd_\Theta}}\right) + 4\log\frac{1}{\delta}}{3D_i Nk}} \tag{100}$$

$$= \sqrt{\frac{4d_\Theta \log\left(\frac{18\sqrt{3}\zeta Nk^2}{16\delta\sqrt{d_\Theta}}\left(L_{2,J} + (GTL_{1,\pi} + G_2)\sqrt{\frac{3}{4}DNk}\right)\right) + 4\log\frac{1}{\delta}}{3D_i Nk}} \tag{101}$$

$$= \sqrt{\frac{4d_\Theta \log\left(\frac{18\sqrt{3}L_{2,J}\zeta Nk^2}{16\delta\sqrt{d_\Theta}} + (GTL_{1,\pi} + G_2)\frac{27\zeta D^{1/2}N^{3/2}k^{5/2}}{16\delta\sqrt{d_\Theta}}\right) + 4\log\frac{1}{\delta}}{3D_i Nk}}. \tag{102}$$

Now, introducing the term:

$$L_1 \coloneqq GTL_{1,\pi} + G_2, \tag{103}$$

we have:

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left((18\sqrt{3}L_{2,J} + 27L_1\sqrt{DNk})\frac{\zeta Nk^2}{16\delta\sqrt{d_\Theta}}\right) + 4\log\frac{1}{\delta}}{3D_i Nk}}. \tag{104}$$

□

D Proofs of Section 5

Theorem 5.1 (RPG Sample Complexity)**.** *Consider to run RPG for $K \in \mathbb{N}$ iterations. Under Assumptions 4.1, 4.2, 4.3, and 4.4, for every $k \in [\![K]\!]$ select the terms $\alpha_{i,k}$ as in Theorem 4.1 and the terms $\lambda_{i,k}$ as:*

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left((18\sqrt{3}L_{2,J} + 27L_1\sqrt{DNk})\frac{\zeta G^2 NKk^2}{8\epsilon\sqrt{d_\Theta}}\right) + 4\log\frac{2G^2K}{\epsilon}}{3D_i Nk}},$$

where $L_1 := GTL_{1,\pi} + G_2$. Selecting a constant step size $\zeta \leqslant 1/L_{2,J}$, with a sample complexity $NK \geqslant \widetilde{\mathcal{O}}(G^2 D d_\Theta \epsilon^{-1})$ and an iteration complexity $K \geqslant \mathcal{O}(\epsilon^{-1})$, it is guaranteed that $\mathbb{E}[\|\nabla J(\boldsymbol{\theta}_{OUT})\|_2^2] \leqslant \epsilon$, where the expectation is taken w.r.t. the learning process and the uniform sampling of $\boldsymbol{\theta}_{OUT}$ from $\{\boldsymbol{\theta}_k\}_{k=0}^{K-1}$.

Proof. For the sake of readability, we divided this proof into four parts. In the first one, we bound the performance difference across subsequent iterations $J(\boldsymbol{\theta}_{k+1}) - J(\boldsymbol{\theta}_k)$. In the second part, we telescope the previous result in order to obtain an upper bound on $\sum_{k=0}^{K-1} \|\nabla J(\boldsymbol{\theta}_k)\|_2^2/K$, for which we employ the result of Theorem 4.2. In the third part, we leverage the result obtained in the second part to compute the convergence rate of RPG, providing a high-probability bound that holds w.p. at least $1-\delta$ for any $\delta \in [0,1]$. Finally, in the last part we select the confidence δ to provide the final result in expectation.

Part (i): Bounding the Performance Difference Across Iterations. Consider to be at iteration $k \in [\![0, K-1]\!]$. Let us start by bounding the difference in performance between $\boldsymbol{\theta}_{k+1}$ and $\boldsymbol{\theta}_k$:

$$J(\boldsymbol{\theta}_{k+1}) - J(\boldsymbol{\theta}_k) \geqslant \langle \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k, \nabla J(\boldsymbol{\theta}_k)\rangle - \frac{L_{2,J}}{2}\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|_2^2 \tag{105}$$

$$= \zeta\left\langle \widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k), \nabla J(\boldsymbol{\theta}_k)\right\rangle - \frac{L_{2,J}}{2}\zeta^2\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)\right\|_2^2, \tag{106}$$

where the first inequality is the quadratic bound holding under Assumption 4.3, while the last inequality follows from the update rule of RPG.

Before going on, note that the following holds for the inner product of $\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)$ and $\nabla J(\boldsymbol{\theta}_k)$:

$$\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k) - \nabla J(\boldsymbol{\theta}_k)\right\|_2^2 = \left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)\right\|_2^2 + \|\nabla J(\boldsymbol{\theta}_k)\|_2^2 - 2\left\langle \widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k), \nabla J(\boldsymbol{\theta}_k)\right\rangle, \tag{107}$$

which implies the following:

$$\left\langle \widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k), \nabla J(\boldsymbol{\theta}_k)\right\rangle = -\frac{1}{2}\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k) - \nabla J(\boldsymbol{\theta}_k)\right\|_2^2 + \frac{1}{2}\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)\right\|_2^2 + \frac{1}{2}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2. \tag{108}$$

By substituting this result into Line (106), for any $k \in [\![0, K-1]\!]$, we obtain the following:

$$J(\boldsymbol{\theta}_{k+1}) - J(\boldsymbol{\theta}_k) \tag{109}$$

$$\geqslant -\frac{\zeta}{2}\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k) - \nabla J(\boldsymbol{\theta}_k)\right\|_2^2 + \frac{\zeta}{2}\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)\right\|_2^2 + \frac{\zeta}{2}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2 - \frac{L_{2,J}}{2}\zeta^2\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)\right\|_2^2 \tag{110}$$

$$= -\frac{\zeta}{2}\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k) - \nabla J(\boldsymbol{\theta}_k)\right\|_2^2 + \frac{\zeta}{2}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2 + \frac{\zeta}{2}(1 - L_{2,J}\zeta)\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k)\right\|_2^2 \tag{111}$$

$$\geqslant -\frac{\zeta}{2}\left\|\widehat{\nabla}^{\text{PM}} J(\boldsymbol{\theta}_k) - \nabla J(\boldsymbol{\theta}_k)\right\|_2^2 + \frac{\zeta}{2}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2, \tag{112}$$

where the last inequality follows by selecting a step size such that $\zeta \leqslant 1/L_{2,J}$.

Part (ii): Telescope the Performance Difference Across Iterations. Now, telescoping the performance difference across iterations $J(\boldsymbol{\theta}_{k+1}) - J(\boldsymbol{\theta}_k)$, the following holds:

$$\sum_{k=0}^{K-1}\left(J(\boldsymbol{\theta}_{k+1}) - J(\boldsymbol{\theta}_k)\right) = J(\boldsymbol{\theta}_K) - J(\boldsymbol{\theta}_0). \tag{113}$$

Moreover, by exploiting the result of Equation (112), the following holds:

$$\sum_{k=0}^{K-1}\left(J(\boldsymbol{\theta}_{k+1})-J(\boldsymbol{\theta}_k)\right)\geqslant-\frac{\zeta}{2}\sum_{k=0}^{K-1}\left\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_k)-\nabla J(\boldsymbol{\theta}_k)\right\|_2^2+\frac{\zeta}{2}\sum_{k=0}^{K-1}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2\,. \tag{114}$$

We recall that in Theorem 4.2 we have the following inequality holding w.p. $1-\delta$ for all $k\in[\![0,K-1]\!]$:

$$\left\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_k)-\nabla J(\boldsymbol{\theta}_k)\right\|_2^2\leqslant\frac{2^8G^2Dd_\Theta}{N(k+1)}\log\left(\frac{6\zeta L_{\text{PM}}N(k+1)^2}{\delta\sqrt{d_\Theta}}\right). \tag{115}$$

Note that, given the chosen notation, the parameterization $\boldsymbol{\theta}_k$ corresponds to a total of $k+1$ iterations. Moreover, for readability purposes, we employed the result of Theorem 4.2 with the term L_{PM}, defined in Lemma C.3, in its implicit form. We further highlight that the expression of L_{PM} of Lemma C.3 still holds even after all the union bounds performed in Theorem 4.2. Please refer to the proof of the latter for a complete explanation of this fact.

Next, we perform a union bound over the iterations K. Before going on with the derivation, we highlight that the terms $\lambda_{i,k}$ become, for any $i\in[\![0,K-1]\!]$ and for any $k\in[\![K]\!]$:

$$\lambda_{i,k}=\sqrt{\frac{4d_\Theta\log\left((18\sqrt{3}L_{2,J}+27L_1\sqrt{DNk})\frac{\zeta NKk^3}{16\delta\sqrt{d_\Theta}}\right)+4\log\frac{K}{\delta}}{3D_iNk}}, \tag{116}$$

where $L_1\coloneqq GTL_{1,\pi}+G_2$.

By performing such a union bound over the iterations K, the previous bound on $\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_k)-\nabla J(\boldsymbol{\theta}_k)\|_2^2$ becomes:

$$\left\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_k)-\nabla J(\boldsymbol{\theta}_k)\right\|_2^2\leqslant\frac{2^8G^2Dd_\Theta}{N(k+1)}\log\left(\frac{6\zeta L_{\text{PM}}NK(k+1)^2}{\delta\sqrt{d_\Theta}}\right) \tag{117}$$

$$\leqslant\frac{2^8G^2Dd_\Theta}{N(k+1)}\log\left(\frac{6\zeta L_{\text{PM}}NK^3}{\delta\sqrt{d_\Theta}}\right). \tag{118}$$

By employing this last result, the following holds w.p. $1-\delta$:

$$\sum_{k=0}^{K-1}\left(J(\boldsymbol{\theta}_{k+1})-J(\boldsymbol{\theta}_k)\right) \tag{119}$$

$$\geqslant-\frac{\zeta}{2}\sum_{k=0}^{K-1}\left\|\widehat{\nabla}^{\text{PM}}J(\boldsymbol{\theta}_k)-\nabla J(\boldsymbol{\theta}_k)\right\|_2^2+\frac{\zeta}{2}\sum_{k=0}^{K-1}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2 \tag{120}$$

$$\geqslant-\frac{2^7G^2Dd_\Theta\zeta}{N}\log\left(\frac{6\zeta L_{\text{PM}}NK^3}{\delta\sqrt{d_\Theta}}\right)\sum_{k=0}^{K-1}\frac{1}{k+1}+\frac{\zeta}{2}\sum_{k=0}^{K-1}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2 \tag{121}$$

$$\geqslant-\frac{2^8G^2Dd_\Theta\zeta\log(K)}{N}\log\left(\frac{6\zeta L_{\text{PM}}NK^3}{\delta\sqrt{d_\Theta}}\right)+\frac{\zeta}{2}\sum_{k=0}^{K-1}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2\,, \tag{122}$$

having exploited $\sum_{k=0}^{K-1}(k+1)^{-1}\leqslant\log(K-1)+1\leqslant2\log(K)$.

Rearranging the previous result and dividing both sides by K, we obtain:

$$\frac{\sum_{k=0}^{K-1}\|\nabla J(\boldsymbol{\theta}_k)\|_2^2}{K}\leqslant\underbrace{\frac{2^9G^2Dd_\Theta\log(K)}{NK}\log\left(\frac{6\zeta L_{\text{PM}}NK^3}{\delta\sqrt{d_\Theta}}\right)}_{=:\mathsf{A}}+\frac{2\left(J(\boldsymbol{\theta}_K)-J(\boldsymbol{\theta}_0)\right)}{\zeta K}. \tag{123}$$

Now, we are going to rearrange the term A for theoretical purposes. By exploiting the fact that $\zeta \leqslant 1/L_{2,J}$ and recovering the full shape of L_{PM} from Lemma C.3, the following holds:

$$\mathsf{A} = \frac{2^9 G^2 D d_\Theta \log(K)}{NK} \log\left(\frac{6\zeta L_{\text{PM}} N K^3}{\delta\sqrt{d_\Theta}}\right) \tag{124}$$

$$\leqslant \frac{2^9 G^2 D d_\Theta \log(K)}{NK} \log\left(\frac{6NK^3 (L_{\text{PM}}/L_{2,J})}{\delta\sqrt{d_\Theta}}\right) \tag{125}$$

$$= \frac{2^9 G^2 D d_\Theta \log(K)}{NK} \log\left(\frac{6NK^3}{\delta\sqrt{d_\Theta}}\left(1 + \frac{GTL_{1,\pi} + G_2}{L_{2,J}}\sqrt{\frac{3}{4}DNK}\right)\right). \tag{126}$$

From now on, for readability purposes, we introduce the following constant:

$$\Psi_1 \coloneqq \frac{GTL_{1,\pi} + G_2}{L_{2,J}}\sqrt{\frac{3}{4}D}. \tag{127}$$

Employing $NK^3 \leqslant NK^3\sqrt{NK}$ and introducing:

$$\Psi_2 \coloneqq \frac{6 + 6\Psi_1}{\delta\sqrt{d_\Theta}}, \tag{128}$$

the following holds:

$$\mathsf{A} \leqslant \frac{2^9 G^2 D d_\Theta \log(K)}{NK} \log\left(\frac{6NK^3}{\delta\sqrt{d_\Theta}}\left(1 + \Psi_1\sqrt{NK}\right)\right) \tag{129}$$

$$\leqslant \frac{2^9 G^2 D d_\Theta}{NK} \underbrace{\log(K) \log\left(\Psi_2 NK^3\sqrt{NK}\right)}_{=:\mathsf{B}}. \tag{130}$$

Focusing on the term B, we have:

$$\mathsf{B} = \log(K) \log\left(\Psi_2 NK^3\sqrt{NK}\right) \tag{131}$$

$$= \log(K)\log(\Psi_2) + \frac{3}{2}\log(K)\log(NK) + 2\log(K)^2 \tag{132}$$

$$\leqslant \left(\frac{7}{2} + \log(\Psi_2)\right)\log(NK)^2 \tag{133}$$

$$\leqslant \Psi_3 \log(NK)^2, \tag{134}$$

being

$$\Psi_3 \coloneqq \frac{7}{2} + \log(\Psi_2). \tag{135}$$

Putting all together, we have the following bound for the term A:

$$\mathsf{A} \leqslant \frac{2^9 G^2 \Psi_3 D d_\Theta \log(NK)^2}{NK}. \tag{136}$$

By plugging the above bound for A into Equation (123), we obtain what follows:

$$\frac{\sum_{k=0}^{K-1} \|\nabla J(\boldsymbol{\theta}_k)\|_2^2}{K} \leqslant \frac{2^9 G^2 \Psi_3 D d_\Theta \log(NK)^2}{NK} + \frac{2\left(J(\boldsymbol{\theta}_K) - J(\boldsymbol{\theta}_0)\right)}{\zeta K} \tag{137}$$

$$\leqslant \frac{2^9 G^2 \Psi_3 D d_\Theta \log(NK)^2}{NK} + \frac{2\left(J^* - J(\boldsymbol{\theta}_0)\right)}{\zeta K}, \tag{138}$$

being $J^* \in \arg\max_{\boldsymbol{\theta}\in\Theta} J(\boldsymbol{\theta})$.

Part (iii)**: Rate Computation.** In order to conclude the proof, we have to find a sample complexity NK such that:

$$\frac{2^9 G^2 \Psi_3 D d_\Theta \log\left(NK\right)^2}{NK} + \frac{2\left(J^* - J(\boldsymbol{\theta}_0)\right)}{\zeta K} \leqslant \frac{\epsilon}{2}. \tag{139}$$

To guarantee the convergence to an ϵ-approximate stationary point, we need two separate conditions, one holding for the iteration complexity K and the other one for the sample complexity NK. The former can be obtained by solving:

$$\frac{2\left(J^* - J(\boldsymbol{\theta}_0)\right)}{\zeta K} \leqslant \frac{\epsilon}{4} \tag{140}$$

$$\implies K \geqslant \frac{8\left(J^* - J(\boldsymbol{\theta}_0)\right)}{\zeta \epsilon} = \mathcal{O}\left(\epsilon^{-1}\right), \tag{141}$$

given that the step size ζ can be any constant such that $\zeta \leqslant 1/L_{2,J}$.

The condition on the sample complexity NK can be obtained by finding the minimum NK such that:

$$\frac{2^9 G^2 \Psi_3 D d_\Theta \log\left(NK\right)^2}{NK} \leqslant \frac{\epsilon}{4}. \tag{142}$$

Alternatively, we can find the maximum NK such that:

$$\frac{2^9 G^2 \Psi_3 D d_\Theta \log\left(NK\right)^2}{NK} \geqslant \frac{\epsilon}{4}. \tag{143}$$

Going for the latter method, and considering that $\log(x)^2 \leqslant 3\sqrt{x}$ for $x \geqslant 1$, we have the following:

$$NK \leqslant \frac{2^{11} G^2 \Psi_3 D d_\Theta \log(NK)^2}{\epsilon} \tag{144}$$

$$\leqslant \frac{2^{11} 3 G^2 \Psi_3 D d_\Theta \sqrt{NK}}{\epsilon} \tag{145}$$

$$\implies NK \leqslant \left(\frac{2^{11} 3 G^2 \Psi_3 D d_\Theta}{\epsilon}\right)^2. \tag{146}$$

Now, switching back to Line (142), we can substitute inside the $\log(NK)$, the term attaining for the minimum value of NK satisfying Line (142) itself:

$$NK \geqslant \frac{2^{11} G^2 \Psi_3 D d_\Theta \log\left(NK\right)^2}{\epsilon} \tag{147}$$

$$\geqslant \frac{2^{13} G^2 \Psi_3 D d_\Theta}{\epsilon} \log\left(\frac{2^{11} 3 G^2 \Psi_3 D d_\Theta}{\epsilon}\right)^2 \tag{148}$$

$$\implies NK \geqslant \widetilde{\mathcal{O}}\left(\epsilon^{-1}\right). \tag{149}$$

We highlight that this sample complexity of order $\widetilde{\mathcal{O}}(\epsilon^{-1})$ is compatible with the iteration complexity $\mathcal{O}(\epsilon^{-1})$ provided in Equation (141), since the considered batch size N is constant.

Part (iv)**: Switching to Expectation.** The last thing to do is to provide a result holding in expectation. In particular, by Jensen's inequality, the following holds:

$$\left\|\nabla J(\boldsymbol{\theta})\right\|_2 = \left\| \mathop{\mathbb{E}}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\mathbf{g}_{\boldsymbol{\theta}}(\tau)\right] \right\|_2 \leqslant \mathop{\mathbb{E}}_{\tau \sim p_{\boldsymbol{\theta}}} \left[\left\|\mathbf{g}_{\boldsymbol{\theta}}(\tau)\right\|_2\right] \leqslant G, \tag{150}$$

having exploited Assumption 4.1. That being said, we have the following:

$$\frac{\sum_{t=1}^K \mathbb{E}\left[\left\|\nabla J(\boldsymbol{\theta}_k)\right\|_2^2\right]}{K} \leqslant \frac{\epsilon}{2} + \delta G^2, \tag{151}$$

whenever $\zeta \leqslant 1/L_{2,J}$, $K \geqslant \mathcal{O}(\epsilon^{-1})$, and $NK \geqslant \widetilde{\mathcal{O}}(\epsilon^{-1})$. By selecting $\delta = \frac{\epsilon}{2G^2}$, we have:

$$\frac{\sum_{t=1}^{K} \mathbb{E}\left[\|\nabla J(\boldsymbol{\theta}_k)\|_2^2\right]}{K} \leqslant \epsilon. \tag{152}$$

We let the reader note that the constant Ψ_3 has the following form:

$$\Psi_3 = \frac{7}{2} + \log\left(\frac{6L_{2,J} + 3\left(GTL_{1,\pi} + G_2\right)\sqrt{3D}}{L_{2,J}\delta\sqrt{d_\Theta}}\right) \tag{153}$$

$$= \frac{7}{2} + \log\left(\frac{6G^2\left(2L_{2,J} + \left(GTL_{1,\pi} + G_2\right)\sqrt{3D}\right)}{L_{2,J}\epsilon\sqrt{d_\Theta}}\right), \tag{154}$$

having substituted the value selected for δ. Finally, we present the explicit shape of the sample complexity:

$$NK \geqslant \frac{2^{13}G^2\Psi_3 D d_\Theta}{\epsilon}\log\left(\frac{2^{11}3G^2\Psi_3 D d_\Theta}{\epsilon}\right)^2 \tag{155}$$

$$\geqslant \frac{2^{13}G^2 D d_\Theta}{\epsilon}\left(\frac{7}{2} + \log\left(\frac{6G^2\left(2L_{2,J} + \left(GTL_{1,\pi} + G_2\right)\sqrt{3D}\right)}{L_{2,J}\epsilon\sqrt{d_\Theta}}\right)\right) \tag{156}$$

$$\cdot \log\left(\frac{2^{11}3G^2 D d_\Theta}{\epsilon}\left(\frac{7}{2} + \log\left(\frac{6G^2\left(2L_{2,J} + \left(GTL_{1,\pi} + G_2\right)\sqrt{3D}\right)}{L_{2,J}\epsilon\sqrt{d_\Theta}}\right)\right)\right)^2, \tag{157}$$

thus being of order:

$$NK \geqslant \widetilde{\mathcal{O}}\left(\frac{G^2 D d_\Theta}{\epsilon}\right). \tag{158}$$

Before concluding the proof, we highlight that the terms $\lambda_{i,k}$ become, for any $i \in [\![0, K-1]\!]$ and for any $k \in [\![K]\!]$, with the selection $\delta = \frac{\epsilon}{2G^2}$:

$$\lambda_{i,k} = \sqrt{\frac{4d_\Theta \log\left(\left(18\sqrt{3}L_{2,J} + 27L_1\sqrt{DNk}\right)\frac{\zeta G^2 NKk^3}{8\epsilon\sqrt{d_\Theta}}\right) + 4\log\frac{2G^2K}{\epsilon}}{3D_i Nk}}, \tag{159}$$

where $L_1 \coloneqq GTL_{1,\pi} + G_2$.

We conclude the proof by noting that by selecting $\boldsymbol{\theta}_{\text{OUT}}$ uniformly at random from the parameterizations encountered during the learning $\{\boldsymbol{\theta}_k\}_{k=0}^{K-1}$, then with the selection of the same sample complexity $NK = \widetilde{\mathcal{O}}(G^2 D d_\Theta \epsilon^{-1})$ and iteration complexity $K = \mathcal{O}(\epsilon^{-1})$, Equation (152) is equivalent to:

$$\mathbb{E}\left[\|\nabla J(\boldsymbol{\theta}_{\text{OUT}})\|_2^2\right] \leqslant \epsilon, \tag{160}$$

where the expectation is taken w.r.t. the entire learning process and the uniform sampling procedure to extract $\boldsymbol{\theta}_{\text{OUT}}$. □

E Implementation Details

In this section, we present the practical version of the RPG method, i.e., the one used in our experimental campaign. Its pseudo-code is provided in Algorithm 2, and it differs from the theoretical version described in Section 3 in the following ways:

i. At iteration k, instead of reusing all previously collected trajectories $\{\{\tau_{i,j}\}_{j=0}^{N-1}\}_{i=0}^{k-1}$ (with corresponding parameterizations $\{\boldsymbol{\theta}_i\}_{i=0}^{k-1}$ such that $\tau_{i,j} \sim p_{\boldsymbol{\theta}_i}$), we retain only the most recent ω iterates, where ω is referred to as the *window size*. This choice improves computational feasibility and mitigates the diminishing utility of older trajectories, which are likely to exhibit greater divergence from the current parameterization $\boldsymbol{\theta}_{k-1}$. A sensitivity analysis on ω is provided in Appendix F.2.

ii. Rather than using the theoretically prescribed values of $\lambda_{i,k}$ and $\alpha_{i,k}$, which depend on the constant D from Assumption 4.2, we adopt adaptive coefficients. Since D is typically unknown in practice, we instead compute $\alpha_{i,k}$ and $\lambda_{i,k}$ based on empirical estimates of the χ^2 divergence between trajectory distributions. These values now reflect the actual discrepancy between sampling and target policies. Details on this estimation procedure are provided in Appendix E.1.

iii. We return the best parameterization observed during training, instead of sampling one uniformly at random from the set of visited iterates. While the theoretical version of RPG relies on uniform sampling to support average-iterate convergence guarantees (see Theorem 5.1), this strategy is not practical.

iv. We replace the constant step size ζ with an adaptive one, optimized via the Adam scheduler (Kingma and Ba, 2015). Section E.2 provides guidance on how to set the initial learning rate when using modern optimizers.

In addition, Section E.3 discusses the differences between the trajectory-based buffer used in RPG and the classic transition-based replay buffer used in actor-critic methods. This practical implementation of RPG is the one employed in the experimental campaign reported in Section 6 and detailed further in Appendix F.

Algorithm 2: RPG (Practical Version).

Input : iterations K, batch size N, learning rate Schedule $\{\zeta_k\}_{i=0}^{K-1}$, initial parameterization $\boldsymbol{\theta}_0$, maximum window length ω, confidence parameter δ

for $k \in [\![0, K-1]\!]$ **do**

Let $\widetilde{k} := \max\{0, k-\omega+1\}$ be the oldest parametrization in the current window.

Collect N trajectories $\{\tau_{k,j}\}_{j=0}^{N-1}$ with policy $\pi_{\boldsymbol{\theta}_k}$.

Estimate the distances between trajectory densities as $\{\widehat{D}_i = \widehat{d}_2(p(\cdot|\theta_k)||p(\cdot|\theta_i))\}_{i=\tilde{k}}^{k}$

Compute $\lambda_{i,k} = \sqrt{\frac{4\log\frac{1}{\delta}}{3\widehat{D}_i N k}}$ and $\alpha_{i,k} = \frac{\widehat{D}_i^{-1/2}}{\sum_{l=\tilde{k}}^{k-1}\widehat{D}_l^{-1/2}}$

Compute the GPOMDP gradient $\mathbf{g}^{\mathrm{G}}_{\boldsymbol{\theta}_k}(\tau_{i,j})$ for each trajectory in the window (i.e., $i \in [\![\tilde{k}, k]\!]$ and $j \in [\![0, N-1]\!]$)

Compute the gradient:

$$\widehat{\nabla}^{\mathrm{PM}} J(\boldsymbol{\theta}_k) = \frac{1}{N}\sum_{i=\tilde{k}}^{k}\sum_{j=0}^{N-1}\frac{\alpha_{i,k}p_{\boldsymbol{\theta}_k}(\tau_{i,j})}{(1-\lambda_{i,k})p_{\boldsymbol{\theta}_i}(\tau_{i,j}) + \lambda_{i,k}p_{\boldsymbol{\theta}_k}(\tau_{i,j})}\mathbf{g}_{\boldsymbol{\theta}_k}(\tau_{i,j})$$

Update the policy parameterization:

$$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \zeta_k\widehat{\nabla}^{\mathrm{PM}} J(\boldsymbol{\theta}_k)$$

end

Return the last parametrization

E.1 Divergence Estimation

Rényi Divergence. Before delving into divergence estimation, we introduce the α-Rényi divergence D_α and its exponentiated version d_α, for any $\alpha \geqslant 1$. Let $P, Q \in \Delta(\mathcal{X})$ admitting densities p and q respectively. If $P \ll Q$, the α-Rényi divergence is defined as:

$$D_\alpha(P\|Q) \coloneqq \frac{1}{\alpha - 1} \log \left(\int p(x)^\alpha q(x)^{1-\alpha} \mathrm{d}x \right). \tag{161}$$

Note that for $\alpha = 1$ we have the KL divergence. The exponentiated α-Rényi divergence is defined as:

$$d_\alpha(P\|Q) \coloneqq \exp\left((\alpha - 1) D_\alpha(P\|Q)\right) = \int p(x)^\alpha q(x)^{1-\alpha} \mathrm{d}x. \tag{162}$$

In the main paper, we employ the χ^2 divergence, which shows the following relation with d_α:

$$\chi^2(P\|Q) = d_2(P\|Q) - 1, \tag{163}$$

thus under Assumption 4.2 it holds the following:

$$\sup_{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \Theta} d_2(p_{\boldsymbol{\theta}_1} \| p_{\boldsymbol{\theta}_2}) \leqslant D. \tag{164}$$

Given the provided equivalence, for the sake of generality and simplicity, we focus on d_α.

Employed Divergence Estimator. In the main manuscript, the convergence result of Theorem 5.1 is established by carefully selecting the sequence of parameters $\alpha_{i,k}$ and $\lambda_{i,k}$, for any $i \in [\![0, K-1]\!]$ and for any $k \in [\![K]\!]$. However, this choice entails two notable drawbacks in practical implementation: (i) the absence of a mechanism to constrain the modification of the parameterization (e.g., a trust-region) makes the determination of the global upper bound D infeasible in practice; (ii) all previous trajectories are treated equally (in terms of $\alpha_{i,k}$ and $\lambda_{i,k}$), regardless of their proximity to the trajectory distribution under the current policy parameter.

To tackle these two problems, we no longer employ the global upper bound D, but we use a dynamic weighting relying on a divergence estimate $\widehat{D}_i \coloneqq \widehat{d}_2(p_{\boldsymbol{\theta}_k}(\cdot) \| p_{\boldsymbol{\theta}_i}(\cdot))$, where $\boldsymbol{\theta}_k$ is the parametrization at the current iteration k and $\boldsymbol{\theta}_i$ is a parametrization belonging to a previous iterate i. The immediate consequence is an increased weighting of trajectories that are collected under "closer" trajectory distributions to the current parametrization, while automatically discarding trajectories generated by "farther" parameterizations. In what follows, we consider two trajectory distributions parameterized by $\boldsymbol{\theta}$ (target) and $\boldsymbol{\theta}_b$ (behavioral).

A naïve estimate of $d_\alpha(p_{\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}_b})$ consists in using the sample mean:

$$\widehat{d}_\alpha(p_{\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}_b}) = \frac{1}{N} \sum_{j=0}^{N-1} \left(\frac{p_{\boldsymbol{\theta}}(\tau_{b,j})}{p_{\boldsymbol{\theta}_b}(\tau_{b,j})} \right)^\alpha, \tag{165}$$

where $\tau_{b,j} \sim p_{\boldsymbol{\theta}_b}$. As one would expect, this estimator is inefficient (Metelli et al., 2018, 2020) and may need a large sample size to be accurate. Empirically, it has also resulted in approximations $\widehat{d}_\alpha(\cdot) \to 0$ violating the positive Rényi divergence constraint $\frac{1}{\alpha - 1} \log(\widehat{d}_\alpha(\cdot)) \geqslant 0$.

A practical $d_\alpha(\cdot)$ estimator has been proposed by (Metelli et al., 2018, 2020), which expresses $\widehat{d}_\alpha(\cdot)$ as a measure of the distance between the two respective parameterized policies at each time step of the trajectory:

$$\widehat{d}_\alpha(p_{\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}_b}) = \frac{1}{N} \sum_{j=0}^{N-1} \prod_{t=0}^{T-1} d_\alpha(\pi_{\boldsymbol{\theta}}(\cdot | \mathbf{s}_{\tau_{b,j},t}) \| \pi_{\boldsymbol{\theta}_b}(\cdot | \mathbf{s}_{\tau_{b,j},t})), \tag{166}$$

where $\tau_{b,j} \sim p_{\boldsymbol{\theta}_b}$. The proposed estimator estimates the distance between two trajectories as the product of the distance of the two policies at each state. The advantage is that this distance can be computed accurately and is not sample-based. However, this depends on the choice of the policy distribution. The properties of this estimator are discussed in greater detail in (Metelli et al., 2020, Remark 6).

Given that our experimental campaign primarily relies on Gaussian policies, as is common practice (Papini et al., 2018; Xu et al., 2019; Yuan et al., 2020; Paczolay et al., 2024), we provide the closed-form expression for $d_\alpha(\pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{s}_{\tau_{b,j},t}) \| \pi_{\boldsymbol{\theta}_b}(\cdot \mid \mathbf{s}_{\tau_{b,j},t}))$ in the case of this kind of policies.

Proposition E.1 (Gil et al. 2013)**.** *Let $\boldsymbol{\theta}, \boldsymbol{\theta}_b \in \Theta$ and let $\mathbf{s} \in \mathcal{S}$. Now, let $\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) = \mathcal{N}(\boldsymbol{\theta}^\top \mathbf{s}, \sigma^2 I_{d_{\mathcal{A}}})$ and $\pi_{\boldsymbol{\theta}_b}(\cdot|\mathbf{s}) = \mathcal{N}(\boldsymbol{\theta}_b^\top \mathbf{s}, \sigma^2 I_{d_{\mathcal{A}}})$, be two $d_{\mathcal{A}}$-dimensional Gaussian policies. Then, it holds:*

$$d_\alpha(\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) \| \pi_{\boldsymbol{\theta}_b}(\cdot|\mathbf{s})) = \exp\left(\frac{\alpha(\alpha-1)\|\boldsymbol{\mu} - \boldsymbol{\mu}_b\|_2^2}{2\sigma^2}\right), \tag{167}$$

where $\boldsymbol{\mu} \coloneqq \boldsymbol{\theta}^\top \mathbf{s}$ and $\boldsymbol{\mu}_b \coloneqq \boldsymbol{\theta}_b^\top \mathbf{s}$.

Proof. We start the proof by recalling the explicit form of $d_\alpha(\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) \| \pi_{\boldsymbol{\theta}_b}(\cdot|\mathbf{s}))$:

$$d_\alpha(\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) \| \pi_{\boldsymbol{\theta}_b}(\cdot|\mathbf{s})) = \exp\left((\alpha-1) D_\alpha(\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}) \| \pi_{\boldsymbol{\theta}_b}(\cdot|\mathbf{s}))\right) \tag{168}$$

$$= \int \pi_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{s})^\alpha \pi_{\boldsymbol{\theta}_b}(\mathbf{x} \mid \mathbf{s})^{1-\alpha} \mathrm{d}\mathbf{x}. \tag{169}$$

By exploiting the fact that both $\pi_{\boldsymbol{\theta}}$ and $\pi_{\boldsymbol{\theta}_b}$ are multivariate Gaussian policies, the following derivation holds:

$$\int \pi_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{s})^\alpha \pi_{\boldsymbol{\theta}_b}(\mathbf{x} \mid \mathbf{s})^{1-\alpha} \mathrm{d}\mathbf{x} \tag{170}$$

$$= \int \left[\frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\tfrac{1}{2\sigma^2}\|\mathbf{x} - \boldsymbol{\mu}\|_2^2\right)\right]^\alpha \left[\frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\tfrac{1}{2\sigma^2}\|\mathbf{x} - \boldsymbol{\mu}_b\|_2^2\right)\right]^{1-\alpha} \mathrm{d}\mathbf{x} \tag{171}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\frac{\alpha\|\mathbf{x} - \boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\mathbf{x} - \boldsymbol{\mu}_b\|_2^2}{2\sigma^2}\right) \mathrm{d}\mathbf{x} \tag{172}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\Big(-\frac{1}{2\sigma^2}\left(\alpha\left(\|\mathbf{x}\|_2^2 - 2\langle\boldsymbol{\mu}, \mathbf{x}\rangle + \|\boldsymbol{\mu}\|_2^2\right)\right. \tag{173}$$

$$\left. + (1-\alpha)\left(\|\mathbf{x}\|_2^2 - 2\langle\boldsymbol{\mu}_b, \mathbf{x}\rangle + \|\boldsymbol{\mu}_b\|_2^2\right)\right)\Big) \mathrm{d}\mathbf{x} \tag{174}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\Big(-\frac{1}{2\sigma^2}\Big(\|\mathbf{x}\|_2^2 - 2\left(\alpha\boldsymbol{\mu} + (1-\alpha)\boldsymbol{\mu}_b\right)^\top \mathbf{x} \tag{175}$$

$$+ \alpha\|\boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2\Big)\Big) \mathrm{d}\mathbf{x}. \tag{176}$$

Now, by letting $\boldsymbol{\mu}_\alpha = \alpha\boldsymbol{\mu} - (1-\alpha)\boldsymbol{\mu}_b$, we have the following:

$$\int \pi_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{s})^\alpha \pi_{\boldsymbol{\theta}_b}(\mathbf{x} \mid \mathbf{s})^{1-\alpha} \mathrm{d}\mathbf{x} \tag{177}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\frac{1}{2\sigma^2}\left(\|\mathbf{x}\|_2^2 - 2\left(\alpha\boldsymbol{\mu} + (1-\alpha)\boldsymbol{\mu}_b\right)^\top \mathbf{x} + \alpha\|\boldsymbol{\mu}\|^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2\right)\right) \mathrm{d}\mathbf{x} \tag{178}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\frac{1}{2\sigma^2}\left(\|\mathbf{x}\|_2^2 - 2\langle\boldsymbol{\mu}_\alpha, \mathbf{x}\rangle + \alpha\|\boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2\right)\right) \mathrm{d}\mathbf{x}. \tag{179}$$

Adding and subtracting $\|\boldsymbol{\mu}_\alpha\|_2^2$ inside the exponent:

$$\int \pi_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{s})^\alpha \pi_{\boldsymbol{\theta}_b}(\mathbf{x} \mid \mathbf{s})^{1-\alpha} \mathrm{d}\mathbf{x} \tag{180}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\frac{1}{2\sigma^2}\left(\|\mathbf{x} - \boldsymbol{\mu}_\alpha\|_2^2 + \alpha\|\boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2 - \|\boldsymbol{\mu}_\alpha\|_2^2\right)\right) \mathrm{d}\mathbf{x} \tag{181}$$

$$= \int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \boldsymbol{\mu}_\alpha\|_2^2\right) \underbrace{\exp\left(-\frac{1}{2\sigma^2}\left(\alpha\|\boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2 - \|\boldsymbol{\mu}_\alpha\|_2^2\right)\right)}_{=:\mathsf{A}} \mathrm{d}\mathbf{x}. \tag{182}$$

Now that A is independent of $\mathbf{x}$, we continue our derivation as:

$$\int \pi_{\boldsymbol{\theta}}(\mathbf{x} \mid \mathbf{s})^{\alpha} \pi_{\boldsymbol{\theta}_b}(\mathbf{x} \mid \mathbf{s})^{1-\alpha} \mathrm{d}\mathbf{x} \tag{183}$$

$$= \exp\left(-\frac{1}{2\sigma^2}\left(\alpha\|\boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2 - \|\boldsymbol{\mu}_\alpha\|_2^2\right)\right) \cdot \underbrace{\int \frac{1}{(2\pi)^{d_{\mathcal{A}}/2}\sigma} \exp\left(-\frac{\|\mathbf{x}-\boldsymbol{\mu}_\alpha\|_2^2}{2\sigma^2}\right) \mathrm{d}\mathbf{x}}_{=1} \tag{184}$$

$$= \exp\left(-\frac{1}{2\sigma^2}\left(\alpha\|\boldsymbol{\mu}\|_2^2 + (1-\alpha)\|\boldsymbol{\mu}_b\|_2^2 - (\alpha^2\|\boldsymbol{\mu}\|_2^2 + 2\alpha(1-\alpha)\langle\boldsymbol{\mu}, \boldsymbol{\mu}_b\rangle + (1-\alpha)^2\|\boldsymbol{\mu}_b\|_2^2)\right)\right) \tag{185}$$

$$= \exp\left(-\frac{1}{2\sigma^2}\left(\alpha(1-\alpha)\|\boldsymbol{\mu}\|_2^2 - 2\alpha(1-\alpha)\langle\boldsymbol{\mu}, \boldsymbol{\mu}_b\rangle + \alpha(1-\alpha)\|\boldsymbol{\mu}_b\|_2^2\right)\right) \tag{186}$$

$$= \exp\left(-\frac{\alpha(1-\alpha)\|\boldsymbol{\mu}-\boldsymbol{\mu}_b\|_2^2}{2\sigma^2}\right) \tag{187}$$

$$= \exp\left(\frac{\alpha(\alpha-1)\|\boldsymbol{\mu}-\boldsymbol{\mu}_b\|_2^2}{2\sigma^2}\right), \tag{188}$$

which concludes the proof. □

E.2 RPG's Behavior with Modern Optimizers

The magnitude of $\widehat{D}_i$ is intrinsically linked to the step size of the parameter updates. Indeed, since large policy deviations from the current parameterization may incur in penalties for scoring the seen trajectories, employing a large step size may cause the update to depend almost exclusively on the most recent batch of trajectories. This phenomenon bears analogy to the behavior of step size schedulers such as Adam (Kingma and Ba, 2015). Specifically, when Adam exhibits uncertainty regarding the gradient's direction, it reduces the effective learning rate, thereby placing greater emphasis on accumulated gradient history and facilitating escape from local optima by integrating information across numerous trajectories. Conversely, when the optimizer attains high directional confidence, manifested as a larger step size, the update is dominated by the information contained in the latest trajectories. Since the estimates of the distance between parameterizations $\widehat{D}_i$ introduce variance, we would like to keep the updates small enough such that old parameterizations are not discarded by small variations in the learning rate, which occurs if the initial learning rate ζ_0 is small enough.

E.3 Analogy with Transition-based Replay Buffers of Actor-Critic Methods

Another widely used variance reduction mechanism in PG methods involves the use of *critics* (Sutton and Barto, 2018). In particular, Actor-Critic methods (Duan et al., 2016) jointly learn a parametric critic (e.g., value or advantage function) alongside the policy (actor). The critic's estimates of value or advantage functions reduce the variance of the policy gradient. This approach has given rise to several successful deep RL algorithms (Mnih et al., 2013; Schulman et al., 2015; Duan et al., 2016), which often rely on experience replay buffers (Lin, 1992). These buffers store individual environment interactions, allowing the agent to sample past transitions at random. In doing so, they effectively smooth the training distribution across multiple past behaviors, decoupling data collection from policy updates.

While RPG also maintains a buffer, there are two key differences: (i) the learning signal is the full cumulative return of each trajectory, rather than individual transitions; and (ii) the entire buffer is used at every iteration, with each trajectory's contribution weighted according to the representativeness of its behavioral policy.

F Experimental details

In this section, we report the hyperparameter configurations used in the experiments presented in the main manuscript, along with additional experiments aimed at validating the empirical performance of RPG. All experiments are conducted using environments from the MuJoCo control suite (Todorov et al., 2012). Table 2 summarizes the observation and action space dimensions, as well as the horizon and discount factor used for each environment. For baseline comparisons, we consider the following methods, already introduced in Section 1:

- GPOMDP (Baxter and Bartlett, 2001);
- SVRPG (Papini et al., 2018);
- SRVRPG (Xu et al., 2019);
- STORM-PG (Yuan et al., 2020);
- DEF-PG (Paczolay et al., 2024).

The code for RPG and GPOMDP is available at `https://github.com/MontenegroAlessandro/MagicRL/tree/offpolicy`. The implementation of DEF-PG was obtained from the original GitHub repository (`https://github.com/paczyg/defpg`), while the remaining baselines were implemented using the Potion library (`https://github.com/T3p/potion`).

Environment Name	Observation Space	Action Space	Horizon	Disc. Factor
Continous Cart Pole (Barto et al., 1983)	$d_{\mathcal{S}} = 4$	$d_{\mathcal{A}} = 1$	$T = 200$	$\gamma = 1$
HalfCheetah-v4 (Todorov et al., 2012)	$d_{\mathcal{S}} = 17$	$d_{\mathcal{A}} = 6$	$T = 100$	$\gamma = 1$
Swimmer-v4 (Todorov et al., 2012)	$d_{\mathcal{S}} = 8$	$d_{\mathcal{A}} = 2$	$T = 200$	$\gamma = 1$

Table 2: Summary of the environments' characteristics.

F.1 Employed Policies

Linear Gaussian Policy: a linear parametric gaussian policy $\pi_{\boldsymbol{\theta}} : \mathcal{S} \to \Delta(\mathcal{A})$ with $d_\Theta = d_{\mathcal{S}} \times d_{\mathcal{A}}$ and with fixed variance σ^2 draws action $\mathbf{a} \sim \mathcal{N}(\boldsymbol{\theta}^\top \mathbf{s}, \sigma^2 I_{d_{\mathcal{A}}})$, being $\mathbf{s} \in \mathcal{S}$ and $\mathbf{a} \in \mathcal{A}$. The score of the policy is defined as follows:

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}) = \frac{((\boldsymbol{a} - \boldsymbol{\theta}^\top \boldsymbol{s})\boldsymbol{s}^\top)^\top}{\sigma^2}. \tag{189}$$

Deep Gaussian Policy: a deep parametric gaussian policy $\pi_{\boldsymbol{\theta}} : \mathcal{S} \to \Delta(\mathcal{A})$ with fixed variance σ^2 draws action $\mathbf{a} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{s}), \sigma^2 I_{d_{\mathcal{A}}})$, where $\mathbf{s} \in \mathcal{S}$, $\mathbf{a} \in \mathcal{A}$, and $\mu_{\boldsymbol{\theta}}(\mathbf{s})$ is the action mean output from the neural network. The score of the policy is the gradient w.r.t. the log probability of the chosen action.

F.2 Window Sensitivity

In this experiment, we study the sensitivity of RPG to the window size ω. Here we conduct the evaluations in the *Continuous Cart Pole* environment (Barto et al., 1983). All learning rates are managed by the Adam (Kingma and Ba, 2015) optimizer, with a starting learning rate of $\zeta_0 = 0.01$. Parameters are initialized by following a standard normal distribution. Exploration is managed by a variance of $\sigma^2 = 0.3$ in the context of linear gaussian policies. RPG uses a fixed batch size of $N = 5$ and we evaluate the performance over a window size $\omega \in \{2, 4, 8, 16, 32\}$ averaged over 10 trials.

As shown in Figure 5, the benefit of increasing the window size ω exhibits diminishing returns: while performance improves significantly when increasing ω from 2 to 8, larger windows (e.g., $\omega = 16$ or beyond) offer no noticeable advantage.

Although this contrasts with the theoretical guarantees, it is consistent with practical expectations for two reasons. First, in RPG, the variance of each term in the gradient estimate is controlled by the divergence $\widehat{D}_i$ between the behavior policy and the current policy (Metelli et al., 2021). Increasing the window size incorporates trajectories from policies that are farther from the current one, thereby increasing variance. Indeed, we recall that we are estimating such $\widehat{D}_i$ terms appearing

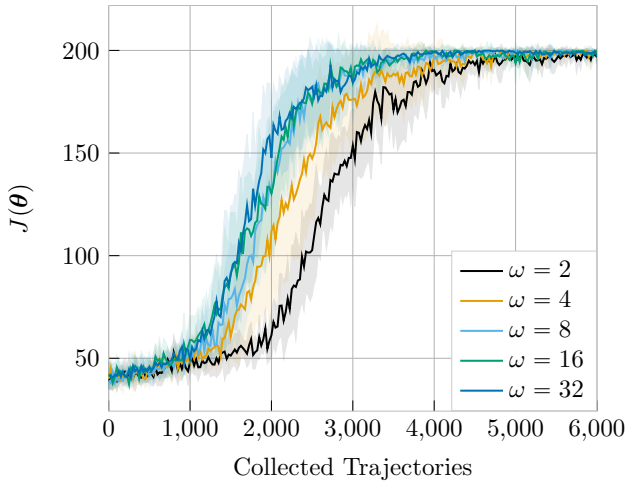


Figure 5: Window sensitivity study of RPG on *Cart Pole* (Appendix F.2). 10 trials (mean $\pm 95\%$ C.I.).

in the method's coefficients (see Appendix E). Second, expanding the window may yield limited additional information, as the data within a smaller window may already be sufficient to capture the correct gradient direction. Moreover, we highlight that a larger ω leads to a larger computational time, since the method is required to evaluate the gradient and the IWs for ωN trajectories.

Consequently, there exists an optimal window size ω that balances variance, information gain, and computational time. This value is generally environment-dependent and challenging to tune in practice. For our experimental campaign, we found that $\omega = 8$ offers a good tradeoff between computational efficiency and performance.

F.3 On Reusing Trajectories

While this experiment was briefly discussed in Section 6, we provide here a more detailed analysis.

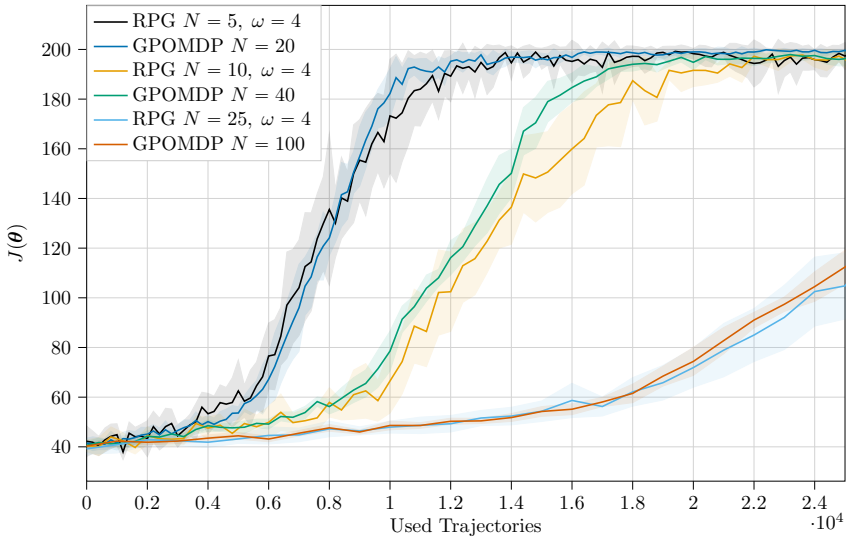
We examine the sensitivity of RPG to the batch size N and compare it with GPOMDP under an equal total data budget, that is, when $\omega N_{\text{RPG}} = N_{\text{GPOMDP}}$, where N_{RPG} is the batch size used by RPG and N_{GPOMDP} the one used by GPOMDP. The aim is to empirically assess the relative informational value of older trajectories versus those collected under the current policy. Specifically, we investigate whether reusing past trajectories, thereby reducing the need for newly sampled data, can accelerate learning in practice.

The evaluations are conducted in the *Continuous Cart Pole* environment (Barto et al., 1983). All methods are trained using the Adam optimizer (Kingma and Ba, 2015), with initial learning rates reported in Table 3.

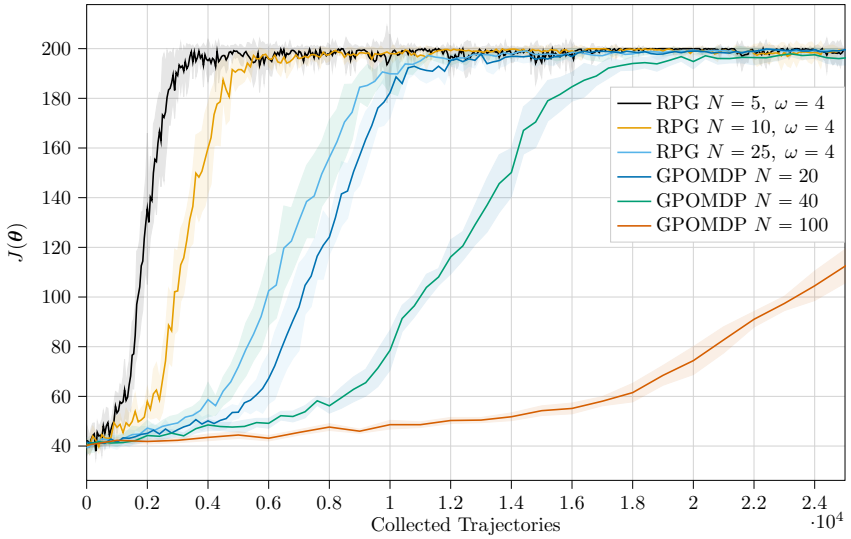
Hyperparameter	TRPG	GPOMDP
Adam ζ_0	0.01	0.01
Parameter Initialization $\boldsymbol{\theta}_0$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$
Variance σ^2	0.3	0.3
Batch size N	$\{5, 10, 25\}$	$\{20, 40, 100\}$
Window Size ω	4	–
Confidence δ	0.05	–

Table 3: Hyperparameters for the experiment in Appendix F.3.

As shown in Figure 6a, when GPOMDP and RPG are matched by number of updates (e.g., RPG with $\omega = 4$ and $N = 5$ versus GPOMDP with $N = 20$), their learning curves are nearly indistinguishable.



(a) RPG vs. GPOMDP w.r.t. Used Trajectories.



(b) RPG vs. GPOMDP w.r.t. Collected Trajectories.

Figure 6: Trajectory reusing study on *Cart Pole* (Appendix F.3). 10 trials (mean $\pm 95\%$ C.I.).

This trend is consistent across various parameter configurations and is supported by statistically significant results.

However, when performance is instead plotted against the number of collected trajectories (Figure 6b), RPG consistently demonstrates faster convergence than GPOMDP across all settings. This provides empirical confirmation that reusing trajectories enhances learning efficiency in practice.

In relatively simple control tasks, previously collected trajectories appear to provide nearly the same informational value as freshly sampled ones, an insight particularly valuable in data-scarce or expensive environments. Additionally, due to its ability to continuously leverage past data, RPG exhibits superior sample efficiency. For instance, GPOMDP with a batch size of $N = 100$ fails to converge within the allowed trajectory budget, while RPG with $\omega = 4$ and $N = 25$ converges rapidly to the optimal policy.

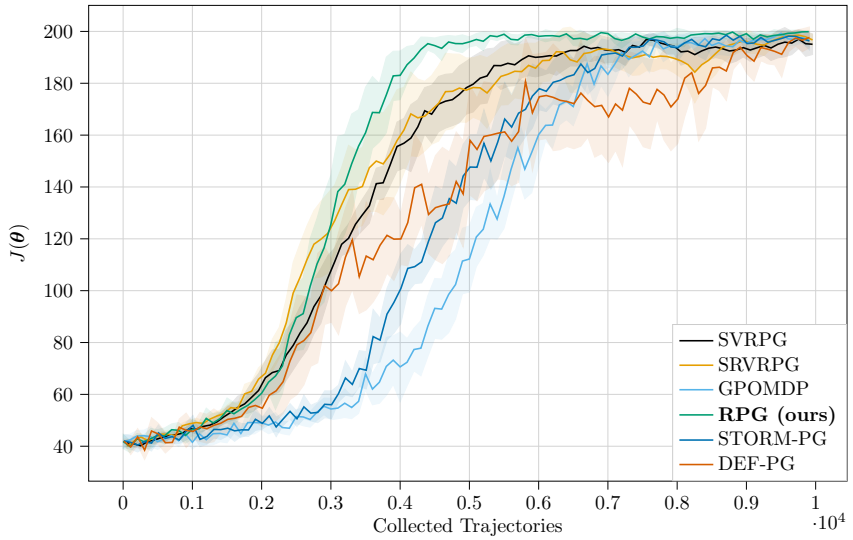


Figure 7: Baselines comparison in *Cart Pole* (Appendix F.4). 10 trials (mean $\pm 95\%$ C.I.).

F.4 Comparison Against Baselines in *Cart Pole*

In this section, we compare our method RPG against PG methods with state-of-the-art rates, whose are discussed in Appendix F. All the methods employ a linear Gaussian policy. We conduct the evaluations in the *Continuous Cart Pole* (Barto et al., 1983) environment. All learning rates are managed by the Adam (Kingma and Ba, 2015) optimizer. All the hyperparameters are specified in Table 4.

Hyperparameter	RPG	GPOMDP	STORM-PG	SRVRPG	SVRPG	DEF-PG
Adam ζ_0	0.01	0.01	0.01	0.01	0.01	0.01
Parameter Initialization $\boldsymbol{\theta}_0$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$	$\mathcal{N}(\mathbf{0}_{d_\Theta}, I_{d_\Theta})$
Variance σ^2	0.3	0.3	0.3	0.3	0.3	0.3
Number of trials	10	10	10	10	10	10
Batch size N	10	10	10	–	–	–
Init-batch size N_0	–	–	10	–	–	–
Window size ω	8	–	–	–	–	–
Confidence δ	0.05	–	–	–	–	–
Snapshot batch size	–	–	–	55	55	55
Mini-batch size	–	–	–	5	5	5

Table 4: Hyperparameters for the experiment in Appendix F.4.

In relatively simple environments like the one considered here, the number of updates plays a crucial role in determining convergence. Therefore, the batch sizes and related parameters are configured to ensure that all methods use the same number of trajectories per update on average. Specifically, SVRPG and SRVRPG perform a full gradient update using a snapshot batch size of 55 every 10^{th} iteration, and stochastic mini-batch updates in between. DEF-PG, a defensive variant of PAGE-PG, performs full gradient updates with probability $p = 0.1$ using the snapshot batch size, and uses mini-batch updates otherwise. As a result, SVRPG, SRVRPG, and DEF-PG each consume an average of 10 trajectories per iteration.

As shown in Figure 7, when evaluated under equal trajectory budgets per iteration (i.e., by matching batch sizes across methods), RPG consistently matches or outperforms all competing baselines and converges more rapidly to the optimal policy. Specifically, in the considered environment, RPG achieves the highest mean return across all trajectory budgets and requires fewer iterations to reach optimal performance, thus confirming that the reuse of past trajectories enables faster convergence.

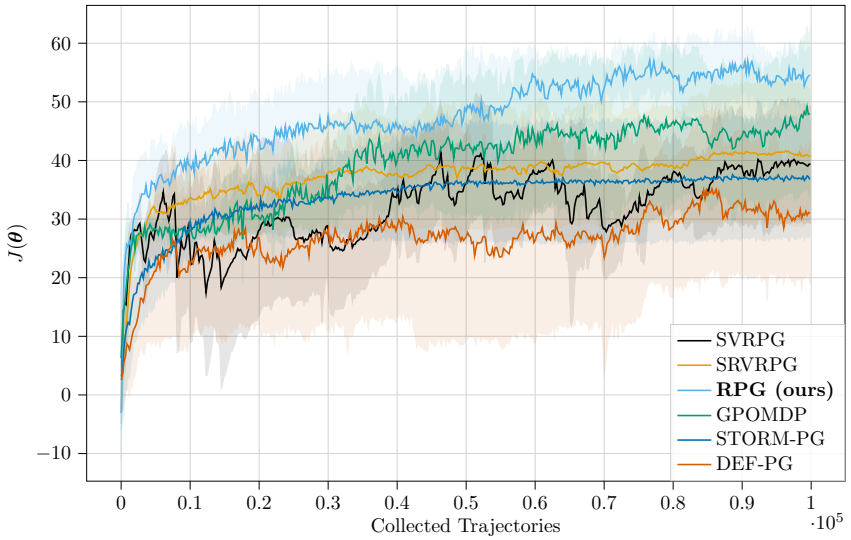


Figure 8: Baselines comparison in *Swimmer* (Appendix F.5). 5 trials (mean $\pm 95\%$ C.I.).

By contrast, STORM-PG and GPOMDP yield nearly overlapping learning curves, indicating comparable sample efficiency under this configuration. DEF-PG, however, exhibits pronounced return oscillations and often fails to sustain monotonic improvement, suggesting instability in the gradient estimates when operating with the same data budget. This behavior indicates that DEF-PG may require larger batch sizes to ensure stable updates.

F.5 Baselines Comparison in *Swimmer*

We conduct the evaluations in the *Swimmer-v4* environment, part of the MuJoCo control suite (Todorov et al., 2012). Using a horizon of $T = 200$ and a discount factor of $\gamma = 1$, this environment is known for featuring a strong local optimum around $J(\boldsymbol{\theta}) \approx 30$. All methods are trained using the Adam optimizer (Kingma and Ba, 2015), with initial learning rates specified in Table 5. Policies are implemented as deep Gaussian networks.

Hyperparameter	**RPG**	**GPOMDP**	**STORM-PG**	**SRVRPG**	**SVRPG**	**DEF-PG**
NN Dimensions	32×32	32×32	32×32	32×32	32×32	32×32
NN Activations	`tanh`	`tanh`	`tanh`	`tanh`	`tanh`	`tanh`
Adam ζ_0	$1e-3$	$1e-3$	$1e-4$	$1e-4$	$1e-3$	$1e-3$
Parameter Initialization $\boldsymbol{\theta}_0$	Xavier	Xavier	Xavier	Xavier	Xavier	Xavier
Variance σ^2	0.3	0.3	0.3	0.3	0.3	0.3
Number of trials	5	5	5	5	5	5
Batch size N	20	20	20	–	–	–
Init-batch size N_0	–	–	20	–	–	–
Window size ω	4	–	–	–	–	–
Confidence δ	0.2	–	–	–	–	–
Snapshot batch size	–	–	–	110	110	110
Mini-batch size	–	–	–	10	10	10

Table 5: Hyperparameters for the experiment in Appendix F.5.

As in previous experiments, we ensure that all methods observe, on average, the same number of trajectories per iteration. This design enables a fair comparison between RPG and the baseline algorithms in terms of data usage and sample efficiency.

As illustrated in Figure 8, RPG not only achieves higher average returns than all competing methods but also demonstrates a greater capacity to escape the environment's strong local optimum. These results further reinforce the effectiveness of trajectory reuse in accelerating convergence and overcoming suboptimal regions in the policy landscape. Interestingly, all competing baselines yield similar performance, and GPOMDP, despite lacking variance reduction techniques, seems to achieve higher mean performances w.r.t. its variance-reduced counterparts. This observation suggests that methods relying on gradient reuse may require larger batch sizes to realize their theoretical advantages effectively.

Despite these clear advantages, the wider confidence intervals reflect notable variability in the number of iterations required to escape the local optimum. This variance is expected, given the task's reward landscape: in some runs, the policy quickly discovers trajectories that facilitate escape, while in others, longer exploratory sequences are necessary. Importantly, this variability highlights the robustness of RPG, which consistently escapes the local optimum across trials, given sufficient interaction time.

F.6 Baselines Comparison in *Half Cheetah*

We conduct the evaluations in the *Half Cheetah-v4* environment, part of the MuJoCo control suite (Todorov et al., 2012). All methods are trained using the Adam optimizer (Kingma and Ba, 2015), with initial learning rates specified in Table 6. The policies are implemented as deep Gaussian networks. As shown in Table 2, *Half Cheetah* features a significantly more complex observation and action space and is the most challenging of the three environments considered in this work.

Hyperparameter	**RPG**	**GPOMDP**	**STORM-PG**	**SRVRPG**	**SVRPG**	**DEF-PG**
NN Dimensions	32×32	32×32	32×32	32×32	32×32	32×32
NN Activations	`tanh`	`tanh`	`tanh`	`tanh`	`tanh`	`tanh`
Adam ζ_0	$1e-4$	$1e-4$	$1e-4$	$1e-4$	$1e-4$	$1e-4$
Parameter Initialization $\boldsymbol{\theta}_0$	Xavier	Xavier	Xavier	Xavier	Xavier	Xavier
Variance σ^2	0.1	0.1	0.1	0.1	0.1	0.1
Number of trials	10	10	10	10	10	10
Batch size N	40	40	40	–	–	–
Init-batch size N_0	–	–	40	–	–	–
Window size ω	8	–	–	–	–	–
Confidence δ	0.2	–	–	–	–	–
Snapshot batch size	–	–	–	256	256	256
Mini-batch size	–	–	–	16	16	16

Table 6: Hyperparameters for the experiment in Appendix F.6.

As shown in Figure 9, RPG exhibits significantly faster convergence, achieving nearly twice the final performance of all competing baselines. This rapid improvement reflects the algorithm's ability to efficiently leverage informative experiences while maintaining sufficient exploration to avoid premature convergence.

Notably, the lower bound of RPG 's confidence interval at convergence exceeds the upper bounds of all baselines, providing strong statistical evidence of its advantages. This underscores the effectiveness of reusing trajectories from past iterations, not only in accelerating convergence but also in escaping local optima, while reducing the need for additional environment interactions. As for the competing baselines, they exhibit similar behaviors, as indicated by the largely overlapping confidence intervals. Interestingly, GPOMDP, which does not employ any variance reduction technique, achieves higher average performance than some of its variance-reduced counterparts, specifically DEF-PG, SRVRPG, and SVRPG. This observation, as further discussed in Appendix F.5, suggests that methods reusing past gradients may require larger batch sizes to fully realize their theoretical advantages.

These results are particularly meaningful in the context of *Half Cheetah*, a benchmark known for its dense rewards and sensitivity to unstable policy updates. RPG 's strong performance in this complex setting further underscores the value of trajectory reuse in continuous, high-dimensional control tasks.

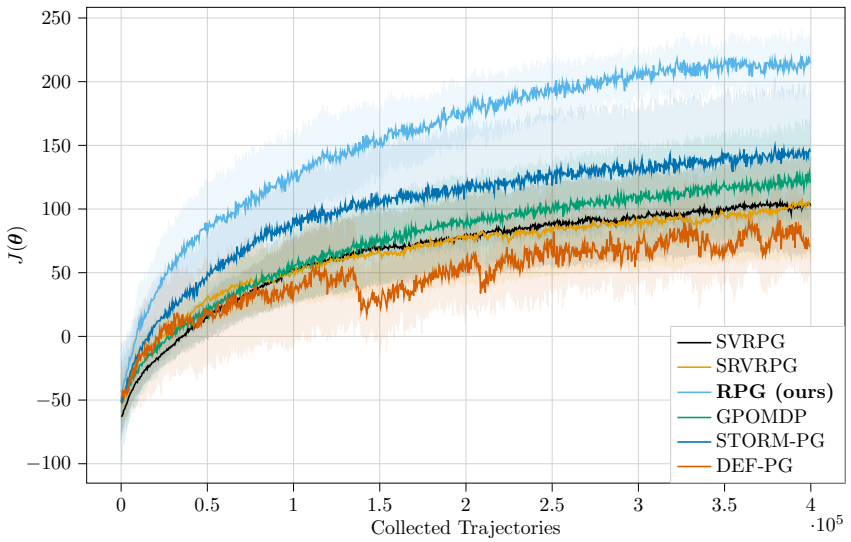


Figure 9: Baselines comparison in *Half-Cheetah* (Appendix F.6). 10 trials (mean $\pm 95\%$ C.I.).

F.7 Computational Resources

All the experiments were run on a machine equipped as follows:

CPU	RAM
AMD Ryzen 7 7800X3D (8 cores, 4.2 GHz)	32 GB 3000 MHz DDR5

Specifically, in the *Cart Pole* environment with a batch size of $N = 100$, a planning horizon of $T = 200$, a linear Gaussian policy, and parallelizing over 10 workers, both the GPOMDP algorithm and RPG (with $\omega = 4$) attain a throughput of ≈ 10 iterations per second. By contrast, when employing a deep Gaussian policy with two hidden layers of 32 units each in the *Half Cheetah* environment ($N = 100$ and $T = 200$), GPOMDP requires ≈ 1.5 seconds per iteration, whereas RPG (with $\omega = 4$) requires ≈ 5.5 seconds per iteration.