A Theoretical Understanding of Gradient Bias in Meta-Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Gradient-based Meta-RL (GMRL) refers to methods that maintain two-level optimisation procedures wherein the outer-loop meta-learner guides the inner-loop gradient-based reinforcement learner to achieve fast adaptations. In this paper, we develop a unified framework that describes variations of GMRL algorithms and points out that existing stochastic meta-gradient estimators adopted by GMRL are actually **biased**. Such meta-gradient bias comes from two sources: 1) the compositional bias incurred by the two-level problem structure, which has an upper bound of $O\big(K\alpha^K\hat{\sigma}_{\text{In}}|\tau|^{-0.5}\big)$ *w.r.t.* inner-loop update step K, learning rate α, estimate variance $\hat{\sigma}^2_{\text{In}}$ and sample size $|\tau|$, and 2) the multi-step Hessian estimation bias $\hat{\Delta}_H$ due to the use of autodiff, which has a polynomial impact $O\big((K-1)(\hat{\Delta}_H)^{K-1}\big)$ on the meta-gradient bias. We study tabular MDPs empirically and offer quantitative evidence that testifies our theoretical findings on existing stochastic meta-gradient estimators. Furthermore, we conduct experiments on Iterated Prisoner's Dilemma and Atari games to show how other methods such as off-policy learning and low-bias estimator can help fix the gradient bias for GMRL algorithms in general.

1 Introduction

Meta Learning, also known as learning to learn, is proposed to equip intelligent agents with meta knowledge for fast adaptations [29]. Meta-Reinforcement Learning (RL) algorithms aim to train RL agents that can adapt to new tasks with only few examples [5, 10, 16]. For example, MAML-RL [10], a typical Meta-RL algorithm, learns the initial parameters of an agent's policy so that the agent can rapidly adapt to new environments with a limited number of policy-gradient updates. Recently, Meta-RL algorithms have been further developed beyond the scope of learning fast adaptations. An important direction is to conduct online meta-gradient learning for adaptively tuning algorithmic hyper-parameters [38, 40] or designing intrinsic reward [41] during training in one single task. Besides, there are Meta-RL developments that manage to discover new RL algorithms by learning algorithmic components or other fundamental concepts in RL, such as the policy gradient objective [26] or the TD-target [39], which can generalise to solve a distribution of different tasks.

In general, gradient-based Meta-RL (GMRL) tasks can be formulated by a two-level optimisation procedure. This procedure optimises the parameters of an outer-loop meta-learner, whose objective is dependent on a K-step policy update process (e.g., stochastic gradient descent) conducted by an inner-loop learner. Formally, this procedure can be written as:

$$\begin{aligned} &\max_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi}) := J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^K), \\ &\text{s.t. } \boldsymbol{\theta}^{i+1} = \boldsymbol{\theta}^i + \alpha \nabla_{\boldsymbol{\theta}^i} J^{\text{In}}(\boldsymbol{\phi}, \boldsymbol{\theta}^i), i \in \{0, 1 \dots K-1\}, \end{aligned} \tag{1}$$

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

where $\boldsymbol{\theta}$ are inner-loop policy parameters, $\boldsymbol{\phi}$ are meta parameters, α is the learning rate, J^{In} and J^{Out} are value functions for the inner and the outer-loop learner. In solving Eq. (1), estimating the meta-gradient of $\nabla_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi})$ from a two-level optimisation process is non-trivial; how to conduct *accurate* estimation on $\nabla_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi})$ has been a critical yet challenging research problem [23, 28, 32].

In this paper, we point out that the meta-gradient estimators adopted by many recent GMRL methods [26, 39] are in fact biased. We conclude that such bias comes from two sources: (1) the **compositional bias** and (2) the **multi-step Hessian bias**. The compositional bias origins from the discrepancy between the sampled policy gradient $\nabla_{\boldsymbol{\theta}} \hat{J}^{\text{In}}$ and expected policy gradient $\nabla_{\boldsymbol{\theta}} J^{\text{In}}$, and the multi-step Hessian estimation bias occurs due to the biased Hessian estimation $\nabla^2_{\boldsymbol{\theta}} \hat{J}^{\text{In}}$ resulting from the employment of automatic differentiation in modern GMRL implementations.

There are very few prior work that investigates the bias in GMRL, most of them limited to the Hessian estimation bias in the MAML-RL setting [28, 32]. Our paper investigates the above two biases in a broader setting and applies on generic meta-gradient estimators. For the compositional bias term, we offer the first theoretical analysis on its quantity, based on which we then investigate current mitigation solutions. For the multi-step Hessian bias, we provide rigorous analysis in GMRL settings particularly for those GMRL tasks where complex inner-loop optimisations are involved.

For the rest of the paper, we first introduce a *unified* Meta-RL framework that can describe variations of existing GMRL algorithms. Building on this framework, we offer two theoretical results that **1)** the compositional bias has an upper bound of $O\big(K\alpha^K \hat{\sigma}_{\text{In}} |\tau|^{-0.5}\big)$ with respect to the inner-loop update step K, the learning rate α, the estimate variance $\hat{\sigma}^2_{\text{In}}$ and the sample size $|\tau|$, and **2)** the multi-step Hessian bias $\hat{\Delta}_H$ has a polynomial impact of $O\big((K-1)(\hat{\Delta}_H)^{K-1}\big)$. To validate our theoretical insights on these two biases, we conduct a comprehensive list ablation studies. Experiments over tabular MDP with MAML-RL [10] and LIRPG [41] demonstrate how quantitatively these two biases influence the estimation accuracy, which consolidates our theories.

Furthermore, our theoretical results help understand to what extent existing methods can mitigate theses two bias empirically. For the compositional bias, we show that off-policy learning methods can reduce the inner-loop policy gradient variance and the resulting compositional bias. For the multi-step Hessian bias, we study how the low-variance curvature technique based on Rothfuss et al. [28] can help correct the Hessian bias for general GMRL problems. We test these solutions on environments including iterated Prisoner's Dilemma with off-policy corrected LOLA-DiCE [13], and eight Atari games based on MGRL [38] with the multi-step Hessian correction technique. Experimental results confirm that those bias-correction methods can substantially decrease the meta-gradients bias and improve the overall performance on rewards.

2 Related Work

A pioneering work that studies meta-gradient estimation is Al-Shedivat et al. [1] who discussed the biased estimation problem of MAML and proposed the E-MAML formulation to fix the meta-gradient bias. Following work includes Rothfuss et al. [28], Liu et al. [23], Tang et al. [32] that tried to fix the meta-gradient estimation error by reducing the estimation variance so as to improve the performance. Moreover, Foerster et al. [12], Farquhar et al. [8], Mao et al. [24] discussed the higher-order gradient estimation in RL. Recently, Bonnet et al. [4], Vuorio et al. [35] proposed algorithms to balance the bias-variance trade-off for meta-gradient estimates. Our work is different from all the above prior work from three folds: (1) we study the additional bias term (the compositional bias) (2) we consider different formulation (expected update while sample-based update in [35], refer to Appendix B for more discussions) (3) we study a broader scope of applications that include different GMRL algorithm instantiations and settings.

In the following part, we review existing GMRL algorithms and categorise them into four research topics. We offer one typical example for each topic in Table 1 and further discussed their relationship and how they can be unified into one framework in Sec. 3. We also provide a more self-contained explanation for each topic in Appendix A for readers that are not familiar with GMRL.

Few-shot RL. The idea of few-shot RL is to enable RL agent with fast learning ability. Specifically, the RL agent is only allowed to interact with the environment for a few trajectory to conduct task-specific adaptation. Conducting few-shot RL has two approaches: gradient based and context based. Context based Meta-RL involves works [5, 15, 27, 36], which uses neural network to embed the

information from few-shot interactions so as to obtain task-relevant context. Gradient based few-shot RL [1, 23, 28] focus on meta-learning the model's initial parameters through meta-gradient descent.

Opponent Shaping. Foerster et al. [13], Letcher et al. [22], Kim et al. [19] explicitly models the learning process of opponents in multi-agent learning problems, which can be thought of as GMRL problems since meta-gradient estimation in these works involve differentiation over opponent's policy updates. By modelling opponents' learning process, the multi-agent learning process can reach better social welfare [13, 22] or the ego agent can adapt to a new peer agent [19].

Single-lifetime Meta-gradient RL. This line of research focuses on learning online adaptation over algorithmic hyper-parameters to enhance the performance of an RL agent in one single task, such as discount factor [38, 40], intrinsic reward generator [41], auxiliary loss [33], reward shaping mechanism [17], and value correction [43]. The main feature of online meta-gradient RL is that they are under single-lifetime setting [39], meaning that the algorithm only iterates through the whole RL learning procedure in one task rather a distribution of tasks.

Multi-lifetime Meta-gradient RL. Compared to the previous single-lifetime settings, "multi-lifetime" refers to settings where agents learn to adapt on a distribution of tasks or environments.The meta-learning target includes policy gradient or TD learning objectives [3, 20, 26], intrinsic reward [42], target value function [39], options in hierarchical RL [34], and recently the design of curriculum in multi-agent learning [9].

3 A Unified Framework for Meta-gradient Estimation

In this section, we derive a general formulation for meta-gradient estimation in GMRL. This formulation enables us to conduct general analysis about meta-gradient estimation and we will show how algorithms in the four research topics mentioned in Sec. 2 can be described through it.

We propose that a general GMRL objective with K-step inner-loop policy gradient update can be written as the following objective

$$\max_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi}) := J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^K), \boldsymbol{\theta}^K = \boldsymbol{\theta}^0 + \alpha \sum_{i=0}^{K-1} \nabla_{\boldsymbol{\theta}^i} J^{\text{In}}(\boldsymbol{\phi}, \boldsymbol{\theta}^i). \tag{2}$$

We denote the meta-parameters as $\boldsymbol{\phi}$, and the pre- and post-adapt inner parameters as $\boldsymbol{\theta}$ and $\boldsymbol{\theta}^K$, respectively. The meta objective as J^{Out}, the inner loop objective as J^{In}. The general objective for GMRL is to maximise the meta objective $J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^K)$, where the inner-loop post-adapt parameters are obtained by taking K policy gradient steps. Note that all inner-loop updates refer to an expected policy gradient (EPG) and all bias term we discuss hereafter is the bias *w.r.t* the **exact** meta-gradient in this K-step EPG inner-loop setting. We discuss the **truncated** K-step EPG setting in Appendix B. The form of exact meta-gradient $\nabla_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi})$ is given by the following proposition via the chain rule.

Proposition 3.1 (K-step Meta-Gradient)**.** *The exact meta-gradient to the objective in Eq.* (2) *can be written as:*

$$\begin{aligned} \nabla_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi}) &= \nabla_{\boldsymbol{\phi}} J^{Out}(\boldsymbol{\phi}, \boldsymbol{\theta}^K) + \alpha \nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^K \nabla_{\boldsymbol{\theta}^K} J^{Out}(\boldsymbol{\phi}, \boldsymbol{\theta}^K), \\ \nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^K &= \sum_{i=0}^{K-1} \nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}^i} J^{In}(\boldsymbol{\phi}, \boldsymbol{\theta}^i) \prod_{j=i+1}^{K-1} \left(I + \alpha \nabla^2_{\boldsymbol{\theta}^j} J^{In}(\boldsymbol{\phi}, \boldsymbol{\theta}^j) \right). \end{aligned} \tag{3}$$

The derivation of the above proposition is in Appendix F.1.

For each topic mentioned in Sec. 2, we pick one GMRL algorithm example to illustrate how they can be fit into this framework. To describe meta-gradient estimation in RL, we start with basic notations. Consider a discrete-time finite horizon Markov Decision Process (MDP) defined by $\langle \mathcal{S}, \mathcal{A}, p, r, \gamma, H \rangle$. At each time step t, the RL agent observes a state $\boldsymbol{s}_t \in \mathcal{S}$, takes an action $\boldsymbol{a}_t \in \mathcal{A}$ based on the policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)$ parametrised with $\boldsymbol{\theta} \in \mathbb{R}^d$, transits to the next state $\boldsymbol{s}_{t+1} \in \mathcal{S}$ according to the transition function $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$ and receives the reward $r(\boldsymbol{s}_t, \boldsymbol{a}_t)$. We define the return $\mathcal{R}(\boldsymbol{\tau}) = \sum_{t=0}^{H} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)$ as the discounted sum of rewards along a trajectory $\boldsymbol{\tau} := (\boldsymbol{s}_0, \boldsymbol{a}_0, \ldots, \boldsymbol{s}_{H-1}, \boldsymbol{a}_{H-1}, \boldsymbol{s}_H)$ sampled by agent policy. The objective for the RL agent is to maximise the expected discounted sum of rewards $V(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p(\tau;\theta)}[\mathcal{R}(\boldsymbol{\tau})]$. Then the RL agent updates parameter $\boldsymbol{\theta}$ using policy gradient given by $\nabla_{\boldsymbol{\theta}} V(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p(\tau;\theta)}\left[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{\tau}) \mathcal{R}(\boldsymbol{\tau})\right]$ where $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{\tau}) = \sum_{t=0}^{H} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)$. For

Table 1: Four typical gradient-based Meta-RL (GMRL) algorithms.

Category	Algorithms	Meta parameter ϕ	Inner parameter θ
Few-shot RL	MAML [10]	Initial Parameter	Initial Parameter
Opponent Shaping	LOLA [13]	Ego-agent Policy	Other-agent Policy
Single-lifetime MGRL	MGRL [38]	Discount Factor	RL Agent Policy
Multi-lifetime MGRL	LPG [26]	LSTM Network	RL Agent Policy

two-agent RL problems, we can extend the MDP to two-agent MDP (or, Stochastic games [30]) defined by $\langle \mathcal{S}_1, \mathcal{S}_2, \mathcal{A}_1, \mathcal{A}_2, P, r_1, r_2, \gamma, H \rangle$, the learning objective for agent i is to maximise its value function of $V_i(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\tau}_i \sim p(\boldsymbol{\tau}_i; \boldsymbol{\theta}_i)} [\mathcal{R}_i(\boldsymbol{\tau}_i)]$.

MAML. Finn et al. [10] optimized over meta initial parameters to maximize the return of one-step adapted policy: $\boldsymbol{\theta}^1 = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} V(\boldsymbol{\theta})$. In MAML-RL, $J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^1)$ degenerates to $V(\boldsymbol{\theta}^1)$ and $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ represent the same initial parameters. The meta-gradient can be derived in the form of Eq. (3): $\nabla_{\boldsymbol{\theta}} \boldsymbol{\theta}^1 \nabla_{\boldsymbol{\theta}^1} V(\boldsymbol{\theta}^1)$, where $\nabla_{\boldsymbol{\theta}} \boldsymbol{\theta}^1 = I + \alpha \nabla^2_{\boldsymbol{\theta}} V(\boldsymbol{\theta})$.

LOLA. Foerster et al. [13] proposed a new learning objective by including an additional term that accounts for the impact of the ego policy to the anticipated opponent's gradient update. For LOLA-agent with parameters $\boldsymbol{\phi}$, it will optimise its return over one-step-lookahead opponent parameters $\boldsymbol{\theta}^1$. The meta-gradient can be shown as: $\nabla_{\boldsymbol{\phi}} V_1(\boldsymbol{\phi}, \boldsymbol{\theta}^1) + \nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1 \nabla_{\boldsymbol{\theta}^1} V_1(\boldsymbol{\phi}, \boldsymbol{\theta}^1)$, where $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1 = \alpha \nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}} V_2(\boldsymbol{\phi}, \boldsymbol{\theta})$

MGRL. Xu et al. [38] proposed to tune the discount factor γ and bootstrapping parameter λ in an online manner. The main feature of MGRL is to conduct inner-loop RL policy $\boldsymbol{\theta}$ update and outer-loop meta parameters $\boldsymbol{\phi} = (\gamma, \lambda)$ alternately. In MGRL, $J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^1)$ degenerates to $V(\boldsymbol{\theta}^1)$. The meta-gradient takes the form: $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1 \nabla_{\boldsymbol{\theta}^1} V(\boldsymbol{\theta}^1)$, $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1 = \alpha \nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}} V(\boldsymbol{\phi}, \boldsymbol{\theta})$.

LPG. Oh et al. [26] aimed to learn a neural network based RL algorithm, by which a RL agent can be properly trained. In LPG, $\boldsymbol{\theta}$ represents the RL agent policy parameters and $\boldsymbol{\phi}$ is the meta-parameter of neural LSTM RL algorithm, $J^{\text{In}}(\boldsymbol{\phi}, \boldsymbol{\theta})$ denotes $f(\boldsymbol{\phi}, \boldsymbol{\theta})$, which is the output of meta-network $\boldsymbol{\phi}$ for conducting inner-loop neural policy gradients. The meta-gradient can be shown as: $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1 \nabla_{\boldsymbol{\theta}^1} V(\boldsymbol{\theta}^1)$, $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1 = \alpha \nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\phi}, \boldsymbol{\theta})$.

Remark. The analytical form of exact meta-gradient given in Eq. (3) involves computation of first-order gradient $\nabla_{\boldsymbol{\theta}} J^{\text{In}}$, $\nabla_{\boldsymbol{\phi}} J^{\text{Out}}$ and$\nabla_{\boldsymbol{\theta}} J^{\text{Out}}$, Jacobian $\nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}} J^{\text{In}}$ and Hessian $\nabla^2_{\boldsymbol{\theta}} J^{\text{In}}$. In practice, these four quantities can be estimated by random samples from inner-loop update step 0 to $K-1$, which denoted by $\boldsymbol{\tau}_0^{0:K-1}, \boldsymbol{\tau}_1^{0:K-1}, \boldsymbol{\tau}_2^{0:K-1}, \boldsymbol{\tau}_3$. As a result, the estimated gradient $\nabla_{\boldsymbol{\phi}} \hat{J}^K(\boldsymbol{\phi})$ can be derived as:

$$
\begin{aligned}
\nabla_{\boldsymbol{\phi}} \hat{J}^K(\boldsymbol{\phi}) &= \nabla_{\boldsymbol{\phi}} \hat{J}^{\text{Out}}(\boldsymbol{\phi}, \hat{\boldsymbol{\theta}}^K, \boldsymbol{\tau}_3) + \alpha \nabla_{\boldsymbol{\phi}} \hat{\boldsymbol{\theta}}^K \nabla_{\hat{\boldsymbol{\theta}}^K} \hat{J}^{\text{Out}}(\boldsymbol{\phi}, \hat{\boldsymbol{\theta}}^K, \boldsymbol{\tau}_3), \\
\nabla_{\boldsymbol{\phi}} \hat{\boldsymbol{\theta}}^K &= \sum_{i=0}^{K-1} \nabla_{\boldsymbol{\phi}} \nabla_{\hat{\boldsymbol{\theta}}^i} J^{\text{In}}(\boldsymbol{\phi}, \hat{\boldsymbol{\theta}}^i, \boldsymbol{\tau}_1^i) \prod_{j=t+1}^{K-1} \left(I + \alpha \nabla^2_{\hat{\boldsymbol{\theta}}^j} J^{\text{In}}(\boldsymbol{\phi}, \hat{\boldsymbol{\theta}}^j, \boldsymbol{\tau}_2^j) \right).
\end{aligned} \tag{4}
$$

The post-adapt inner parameter estimate takes the form $\hat{\boldsymbol{\theta}}^K = \boldsymbol{\theta}^0 + \alpha \sum_{i=0}^{K-1} \nabla_{\hat{\boldsymbol{\theta}}^i} \hat{J}^{\text{In}}(\boldsymbol{\phi}, \hat{\boldsymbol{\theta}}^i, \boldsymbol{\tau}_0^i)$.

4 Theoretical Analysis of Meta-gradient Estimators

In this section, we systematically discuss and theoretically analyse the bias and variance terms for meta-gradient estimations in the current GMRL literature. We highlight two important sources of biases in meta-gradient estimations: the compositional bias and the multi-step Hessian bias. Our analysis builds on the following three assumptions:

Assumption 4.1 (Lipschitz continuity)**.** The outer-loop objective function J^{Out} satisfies that $J^{\text{Out}}(\cdot, \boldsymbol{\theta})$ and $J^{\text{Out}}(\boldsymbol{\phi}, \cdot)$ are Lipschitz continuous with constants $m_{\boldsymbol{\theta}}$ and $m_{\boldsymbol{\phi}}$ respectively, $m_1 = \max_{\boldsymbol{\theta}} m_{\boldsymbol{\theta}}$, $m_2 = \max_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}$. $\nabla_{\boldsymbol{\phi}} J^{\text{Out}}(\cdot, \boldsymbol{\theta})$ and $\nabla_{\boldsymbol{\theta}} J^{\text{Out}}(\boldsymbol{\phi}, \cdot)$ are Lipschitz continuous with constants $\mu_{\boldsymbol{\theta}}$ and $\mu_{\boldsymbol{\phi}}$ respectively, $\mu_1 = \max_{\boldsymbol{\theta}} \mu_{\boldsymbol{\theta}}$, $\mu_2 = \max_{\boldsymbol{\phi}} \mu_{\boldsymbol{\phi}}$. The inner-loop objective function J^{In} satisties that $\nabla_{\boldsymbol{\theta}} J^{\text{In}}(\cdot, \boldsymbol{\theta})$and $\nabla_{\boldsymbol{\theta}} J^{\text{In}}(\boldsymbol{\phi}, \cdot)$ are Lipschitz continuous with constants $c_{\boldsymbol{\theta}}$ and $c_{\boldsymbol{\phi}}$ respectively, $c_1 = \max_{\boldsymbol{\theta}} c_{\boldsymbol{\theta}}$, $c_2 = \max_{\boldsymbol{\phi}} c_{\boldsymbol{\phi}}$. $\nabla^2_{\boldsymbol{\theta}} J^{\text{In}}(\boldsymbol{\phi}, \cdot)$ is Lipschitz continuous with constants $\rho_{\boldsymbol{\phi}}$, $\rho_2 = \max_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}$.

Assumption 4.2 (Bias of estimators)**.** Outer-loop stochastic gradient estimator $\nabla_{\boldsymbol{\phi}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\tau})$ and $\nabla_{\boldsymbol{\theta}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\tau})$ are unbiased estimator of $\nabla_{\boldsymbol{\phi}}J^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta})$ and $\nabla_{\boldsymbol{\theta}}J^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta})$. Inner-loop stochastic gradient estimator $\nabla_{\boldsymbol{\theta}}\hat{J}^{\text{In}}(\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\tau})$ is unbiased estimator of $\nabla_{\boldsymbol{\theta}}J^{\text{In}}(\boldsymbol{\phi},\boldsymbol{\theta})$.

Assumption 4.3 (Variance of estimators)**.** The outer-loop stochastic gradient estimator $\nabla_{\boldsymbol{\phi}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\tau})$ and $\nabla_{\boldsymbol{\theta}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\tau})$ has bounded variance, i.e., $\mathbb{E}_{\tau}[\|\nabla_{\boldsymbol{\phi}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\cdot,\boldsymbol{\tau}) - \nabla_{\boldsymbol{\phi}}J^{\text{Out}}(\boldsymbol{\phi},\cdot)\|^2] \leq (\sigma_1)^2$, and $\mathbb{E}_{\tau}[\|\nabla_{\boldsymbol{\theta}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\cdot,\boldsymbol{\tau}) - \nabla_{\boldsymbol{\theta}}J^{\text{Out}}(\boldsymbol{\phi},\cdot)\|^2] \leq (\sigma_2)^2$.

The above three assumptions are all common ones adopted by existing work [6, 7, 18].

4.1 The Compositional Bias

Recall the K-step inner-loop meta-gradient estimate in Eq. (4), existing GMRL methods usually get unbiased outer-loop gradient estimator $\nabla_{\boldsymbol{\phi}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^K,\boldsymbol{\tau}_3)$ and $\nabla_{\hat{\boldsymbol{\theta}}^K}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^K,\boldsymbol{\tau}_3)$, then the algorithm can get unbiased meta-gradient estimation by plugging in unbiased inner-loop gradient estimation $\hat{\boldsymbol{\theta}}^K$, where $\mathbb{E}[\hat{\boldsymbol{\theta}}^K] = \boldsymbol{\theta}^K$. However, this is not true because of the compositional optimisation structure.

Consider a non-linear compositional scalar objective $f(\boldsymbol{\theta}^K)$, the gradient estimation bias comes from the fact that

$$f(\boldsymbol{\theta}^K) = f(\mathbb{E}[\hat{\boldsymbol{\theta}}^K]) \neq \mathbb{E}[f(\hat{\boldsymbol{\theta}}^K)].$$

If one substitutes the non-linear function $f(\boldsymbol{\theta}^K)$ with $\nabla_{\boldsymbol{\theta}^K}J^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta}^K)$ and $\nabla_{\boldsymbol{\phi}}J^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta}^K)$, then a typical meta-gradient estimation in GMRL introduces compositional bias:

$$\begin{aligned}\mathbb{E}[\nabla_{\hat{\boldsymbol{\theta}}^K}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^K,\tau_3)] &= \mathbb{E}[\nabla_{\hat{\boldsymbol{\theta}}^K}J^{\text{Out}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^K)] \neq \nabla_{\boldsymbol{\theta}^K}J^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta}^K),\\ \mathbb{E}[\nabla_{\boldsymbol{\phi}}\hat{J}^{\text{Out}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^K,\tau_3)] &= \mathbb{E}[\nabla_{\boldsymbol{\phi}}J^{\text{Out}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^K)] \neq \nabla_{\boldsymbol{\phi}}J^{\text{Out}}(\boldsymbol{\phi},\boldsymbol{\theta}^K),\end{aligned} \tag{5}$$

which leads to meta-gradient estimation bias. The following lemma characterises compositional bias.

Lemma 4.4 (Compositional Bias)**.** *Suppose that Assumption 4.1 and 4.2 hold, let $\hat{\Delta}_C = \mathbb{E}[\|f(\hat{\boldsymbol{\theta}}^K) - f(\boldsymbol{\theta}^K)\|]$ be the compositional bias and C_0 the Lipschitz constant of $f(\cdot)$, $|\boldsymbol{\tau}|$ denote number of trajectories used to estimate inner-loop gradient in each inner-loop update step, α the learning rate, then we have,*

$$\hat{\Delta}_C \leq C_0\mathbb{E}[\|\hat{\boldsymbol{\theta}}^K - \boldsymbol{\theta}^K\|] \leq C_0\left((1+\alpha c_2)^K - 1\right)\frac{\hat{\sigma}_{In}}{c_2\sqrt{|\boldsymbol{\tau}|}}, \tag{6}$$

where $\hat{\sigma}_{In} = \max_i \sqrt{\mathbb{V}[\nabla_{\boldsymbol{\theta}^i}\hat{J}^{In}(\boldsymbol{\phi},\boldsymbol{\theta}^i,\boldsymbol{\tau}_0^i)]}$, $i \in \{0,..,K-1\}$.

Proof. See Appendix G.1 for a detailed proof. □

Lemma 4.4 indicates that the compositional bias comes from the inner-loop policy gradient estimate, concerning the learning rate α, the sample size $|\boldsymbol{\tau}|$ and the variance of policy gradient estimator $\hat{\sigma}_{\text{In}}$. This is a fundamental issue in many existing GMRL algorithms [13, 38] since applying stochastic policy gradient update can introduce estimation errors, possibly due to large sampling variance, therefore $\hat{\boldsymbol{\theta}}^K \neq \boldsymbol{\theta}^K$. It also implies that the bias issue becomes more serious under the multi-step formulation since each policy gradient step introduces estimation error, resulting in composite biases.

4.2 The Multi-step Hessian Bias

Recall the analytical form of the exact meta-gradient in Eq. (3), estimating $\nabla_{\boldsymbol{\phi}}\boldsymbol{\theta}^K$ involves computing Hessian $\nabla^2_{\boldsymbol{\theta}^j}J^{\text{In}}(\boldsymbol{\phi},\boldsymbol{\theta}^j)$. In Eq. (4), the Hessian term is estimated by $\nabla^2_{\hat{\boldsymbol{\theta}}^j}J^{\text{In}}(\boldsymbol{\phi},\hat{\boldsymbol{\theta}}^j,\tau_2^j)$ where $j \in \{1,\ldots,K-1\}$. Hessian estimation is a non-trivial problem in GMRL, biased Hessian estimation issue has been brought up in various MAML-RL papers[23, 28, 32], we offer a brief summary in Appendix C due to page limit. Beyond MAML-RL, many recent GMRL work suffers from the same bias due to applying direct automatic differentiation. For example, existing work such as Eq. (3) in Oh et al. [26], Eq. (4) in Zheng et al. [42], Eq. (12), (13), (14) in Xu et al. [38] and Eq. (5), (6), (7) in Xu et al. [39] suffers from this issue. Interestingly, most of them are **coincidentally** unbiased if they only conduct only one-step policy gradient update in the inner-loop. For K-step GMRL when $K = 1$, $\nabla_{\boldsymbol{\phi}}\boldsymbol{\theta}^1$ in meta-gradient writes as:

$$\nabla_{\boldsymbol{\phi}}\boldsymbol{\theta}^1 = \nabla_{\boldsymbol{\phi}}\nabla_{\boldsymbol{\theta}^1}J^{\text{In}}(\boldsymbol{\phi},\boldsymbol{\theta}^1). \tag{7}$$

We can see from Eq. (7) that it would not involve Hessian $\nabla^2 J^{\text{In}}$ computation if $\boldsymbol{\phi} \neq \boldsymbol{\theta}$. To further illustrate, in one-step MGRL, we can show that the estimation of $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^1$ are unbiased because it takes derivatives *w.r.t* meta-parameters $\boldsymbol{\phi} = (\gamma, \lambda)$ which don't have gradient dependency on the trajectory distribution. However, when it takes more than one-step inner-loop policy gradient updates, the meta-gradient estimation will get the hessian estimation bias. As a result, for the reason that when $K > 1$ in K-step GMRL, the $\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^K$ in meta-gradient takes the form:

$$\nabla_{\boldsymbol{\phi}} \boldsymbol{\theta}^K = \sum_{i=0}^{K-1} \nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}^i} J^{\text{In}}\left(\boldsymbol{\phi}, \boldsymbol{\theta}^i\right) \prod_{j=t+1}^{K-1} \left(I + \alpha \nabla^2_{\boldsymbol{\theta}^j} J^{\text{In}}\left(\boldsymbol{\phi}, \boldsymbol{\theta}^j\right)\right). \tag{8}$$

This is the reason why we name it by **multi-step** Hessian bias.

4.3 Theoretical Bias-Variance Analysis

Based on Lemma 4.4 and the discussion in Sec. 4.2, we can derive the upper bound on the bias and variance of the meta-gradient with K-step inner-loop updates.

Theorem 4.5 (Upper bound for the bias and the variance)**.** *Suppose that Assumption 4.1 and 4.2 and 4.3 hold. Let $J_{\boldsymbol{\phi},\boldsymbol{\theta}}$ denote $\nabla_{\boldsymbol{\phi}} \nabla_{\boldsymbol{\theta}} J^{In}$, $H_{\boldsymbol{\theta},\boldsymbol{\theta}}$ denote $\nabla^2_{\boldsymbol{\theta}} J^{In}$, $\hat{\Delta}^K = \|\mathbb{E}[\nabla_{\boldsymbol{\phi}} \hat{J}^K(\boldsymbol{\phi})] - \nabla_{\boldsymbol{\phi}} J^K(\boldsymbol{\phi})\|$ be the meta-gradient estimation bias, set $B = 1 + \alpha c_2$. Then the bound of bias hold:*

$$\hat{\Delta}^K \leq O\Bigg((B + \hat{\Delta}_H)^{K-1} \left(\mathbb{E}[\|\hat{\boldsymbol{\theta}}^K - \boldsymbol{\theta}^K\|] + \hat{\Delta}_J + (K-1)\right)\Bigg). \tag{9}$$

Let $(\hat{\sigma}^K)^2 = \mathbb{V}\left[\nabla_{\boldsymbol{\phi}} \hat{J}^K(\boldsymbol{\phi})\right]$ be the meta-gradient estimation variance, set $V_1 = (1 + \alpha c_2)^2$, $V_2 = 2\alpha^2(m_1^2 + 3\sigma_2^2)$ the estimation variance is given by

$$\begin{aligned}(\hat{\sigma}^K)^2 \leq O\Bigg(&(V_1 + \hat{\Delta}_H^2)^{K-1} \left(\mathbb{E}[\|\hat{\boldsymbol{\theta}}^K - \boldsymbol{\theta}^K\|^2] + (K-1)\right) \\ &+ \left(V_2 + (V_1 + \hat{\Delta}_H^2 + \hat{\sigma}_H^2)^{K-1} - (V_1 + \hat{\Delta}_H^2)^{K-1}\right)(\hat{\Delta}_J^2 + \hat{\sigma}_J^2)\Bigg).\end{aligned} \tag{10}$$

where $\hat{\Delta}_J = \max_{\boldsymbol{\phi}\times\boldsymbol{\theta}} \|\mathbb{E}[\hat{J}_{\boldsymbol{\phi},\boldsymbol{\theta}}] - J_{\boldsymbol{\phi},\boldsymbol{\theta}}\|$. $\hat{\Delta}_H = \max_{\boldsymbol{\theta}} \|\mathbb{E}[\hat{H}_{\boldsymbol{\theta},\boldsymbol{\theta}}] - H_{\boldsymbol{\theta},\boldsymbol{\theta}}\|$. $(\hat{\sigma}_J)^2 = \frac{\max_{\boldsymbol{\phi}\times\boldsymbol{\theta}} \mathbb{V}[\hat{J}_{\boldsymbol{\phi},\boldsymbol{\theta}}]}{|\tau|}$. $(\hat{\sigma}_H)^2 = \frac{\max_{\boldsymbol{\theta}} \mathbb{V}[\hat{H}_{\boldsymbol{\theta},\boldsymbol{\theta}}]}{|\tau|}$.

Proof. See Appendix H.1 for a detailed proof. □

Theorem 4.5 shows that the upper bound of bias and variance consists of two parts: the first term indicates compositional bias $\mathbb{E}[\|\hat{\boldsymbol{\theta}}^K - \boldsymbol{\theta}^K\|]$ in Lemma 4.4, while the second term refers to the second-order estimation bias $(\hat{\Delta}_J, \hat{\Delta}_H)$ and the variance $(\hat{\sigma}_J, \hat{\sigma}_H)$. Several observations can be made based on Theorem 4.5: 1) the compositional bias exists in both upper bound of bias and variance; 2) the multi-step Hessian bias has polynomial impact on the upper bound of bias and variance; 3) most importantly, many existing GMRL algorithms suffer from the compositional bias; moreover, the Hessian bias can significantly increase meta-gradient bias in the multi-step inner-loop setting.

4.4 Understanding Existing Mitigations for Meta-gradient Biases

Based on the above theoretical analysis, we now try to explore and understand how existing methods can handle these two estimation biases.

Off-policy Learning. From the Lemma 4.4 and the discussion in Sec. 4.1, we know that the compositional bias is caused by the estimation error between $\hat{\theta}_t^K$ and θ_t^K. In this case, a simple idea is that one can leverage off-policy learning technique [31] to handle the compositional bias problem by reusing samples $\tau^i_{0:t-1,0}$ together with $\tau^i_{t,0}$ to approximate θ_t^K, we want $\hat{\theta}_t^K$ to stay close to θ_t^K. The intuition behind this correction is to enlarge the sample size $|\boldsymbol{\tau}|$ so that the compositional bias can be minimised according to Lemma 4.4. Specifically, we can apply importance sampling technique to correct the compositional bias, i.e., $\mathbb{E}_{\boldsymbol{\tau}\sim p(\boldsymbol{\tau};\boldsymbol{\theta})}\left[\sum_{t=0}^{H-1} \frac{\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\mu(\boldsymbol{a}_t|\boldsymbol{s}_t)} \mathcal{R}_{\boldsymbol{\phi}}(\boldsymbol{\tau})\right]$. In practice, we also need to

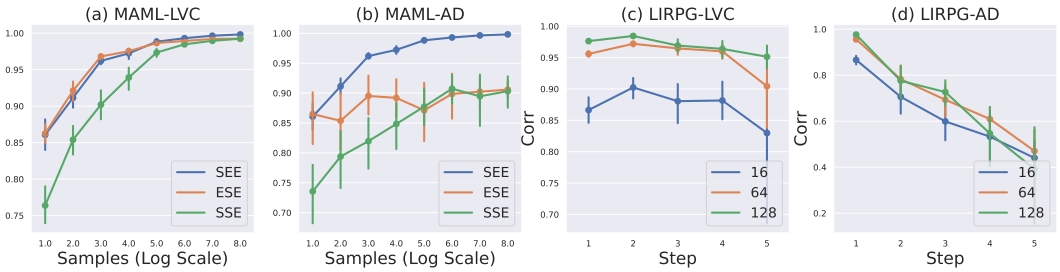

Figure 1: (a, b) Ablation study for sample size and estimators in MAML-RL. “S" is for stochastic estimation while “E" is for exact solution. AD refers to automatic differentiation. (c,d) Ablation study on sample size, steps and estimators in LIRPG.

manually control the level of off-policyness to prevent the variance introduced by the importance sampling process from increasing the estimation variance conversely.

Mutli-step Hessian Estimator. From the theoretical analysis in Sec. 4.3, we can know that the Hessian estimation bias can significantly increase the meta-gradient estimation bias in the multi-step inner-loop setting. Many low-bias hessian estimator have been proposed in the scope of MAML-RL. However, the effect of them have never been verified in the general GMRL context. Here we have a second look at the Low Variance Curvature (LVC) method [28], one can replace the original log-likelihood with the LVC operator in the policy gradient step. As such, the Hessian estimator $\nabla^2_\theta J^{\text{In}}_{\text{LVC}}(\boldsymbol{\phi}, \boldsymbol{\theta})$ takes the form: $\nabla_\theta \mathbb{E}_{\boldsymbol{\tau}\sim p(\boldsymbol{\tau};\boldsymbol{\theta})}\left[\sum_{t=0}^{H-1} \frac{\nabla_{\boldsymbol{\theta}}\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\perp\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_t|\boldsymbol{s}_t)}\mathcal{R}_{\boldsymbol{\phi}}(\boldsymbol{\tau})\right]$ where $\perp$ is the stop-gradient operation which detaches the gradient dependency from the computation graph. As shown in [28], the LVC operator will ensure an unbiased first-order policy gradient and low-biased low-variance second-order policy gradient. Essentially, this operator only corrects the meta-gradient update and leaves the inner-loop gradient estimation formula untouched.

5 Experiments

In this section we conduct empirical evaluation of our proposed bias analysis and the proposed methods to mitigate the biases, and our experiments cover all 3 GMRL fields listed in Table 1. In particular, we conduct a tabular MDP experiment to show the existence of two biases discussed in Sec. 4 using MAML [10] and LIRPG [41]. In order to show how the proposed methods can mitigate the compositional bias, we consider a Iterated Prisoner Dilemma (IPD) problem and use LOLA [13] with off-policy corrections in Sec. 4.4; Similarly, we conduct evaluations on Atari games using MGRL [38] with LVC corrections in Sec. 4.4 to mitigate the Hessian estimation bias.

5.1 Investigate the bias of meta-gradient estimators

Firstly, we conduct experiments to study the bias of meta-gradient estimators using a tabular example using MAML and LIRPG. To align with existing works in the literature, we adopt the settings of random MDPs in [32] with the focus on meta-gradient estimation. We refer readers to Appendix J.1.1 for more experimental details.

MAML-RL. Recall that the meta-gradient in MAML is $\nabla_{\boldsymbol{\theta}^0} J^{\text{Out}}(\boldsymbol{\theta}^K)$. To control the effect of gradient estimations, we use the three estimators to estimate following three terms: (I) inner-loop policy gradient $\nabla_{\boldsymbol{\theta}} J^{\text{In}}(\boldsymbol{\theta})$; (II) Jacobian/Hessian $\nabla^2_{\boldsymbol{\theta}} J^{\text{In}}(\boldsymbol{\theta})$; (III) outer-loop policy gradient $\nabla_{\boldsymbol{\theta}^K} J^{\text{Out}}(\boldsymbol{\theta}^K)$. Refer to Appendix J.1.2 for the implementation of decomposing meta-gradient estimation with different estimators.

To show how the estimation is biased, we use a stochastic estimator denoted as S, and exact analytic calculator denoted as E, for all three derivative terms. Thus, we can have 7 valid permutations in the experiment to validate the estimation, where the rest EEE estimator is the exact gradient.

We conduct our ablation studies by comparing the correlation between of meta-gradient with the exact one. We pick this metric because the bias and variance of meta-gradient are both related with compositional bias and Hessian bias from Theorem 4.5. The correlation metric, which is determined by bias and variance, can show how the final estimation quality is influenced by these two bias terms. As the quality of stochastic estimators vary from many factors, we conduct this ablation study under extensive combinations of estimation algorithms, learning rates, sample sizes, .etc. Due to the page

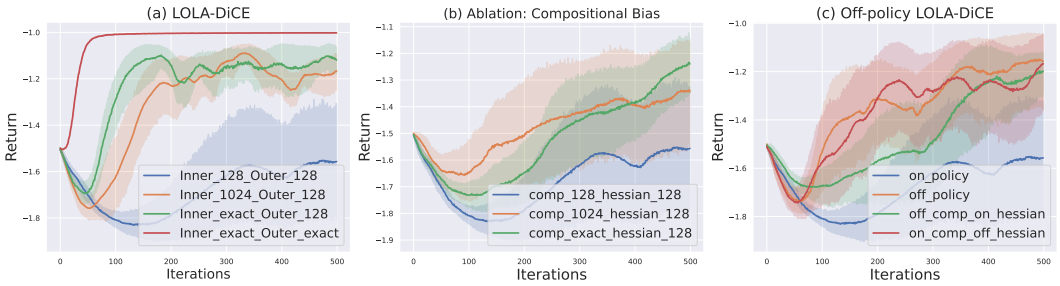

Figure 2: Experiment result of LOLA-DiCE over 10 seeds. The Inner_A_Outer_B legend means we use A samples to estimate inner-loop gradient while B samples to estimate outer-loop gradient. The 'exact' means we use analytical solution of policy gradient instead of estimation.

limit, the results illustrated in Fig. 1(a,b) only include ablation study on sample size and estimation using exact gradients for (III), the outer-loop policy gradient. For the rest ablation study and more evaluation metrics (variance of estimation), refer to Appendix J.1.3 for more details.

We start our evaluation by increasing the sample size. For simplicity, we only conduct one inner step here. In Fig. 1(a), we leverage the LVC [28] estimator. We can see that a correct inner-loop policy estimation and/or a Hessian estimation can significantly improve the estimation quality (as SEE $\approx$ ESE $>$ SSE in low sample size case). By increasing the sample size, we can see the gap between them is shrinking, which verifies the finding in Lemma 4.4. In this case the compositional bias correction shares the same importance with the Hessian bais correction (SEE $\approx$ ESE). We also compare it with the original yet biased gradient estimator of MAML in Fig. 1(b), in which SEE $>$ ESE in all sample size settings. In fact, since the gradient estimation is biased, only SEE achieves near 1.0 correlation provided sufficient samples, again confirms the importance of Hessian bias corrections in Sec. 4.2.

LIRPG. In this setting, we follow the algorithm of intrinsic reward generator presented in [41]. In tabular MDP, we have an additional meta intrinsic reward matrix ϕ. Starting from $\boldsymbol{\theta}^0$, the inner-loop process takes policy gradient based on the new reward matrix $R_{\text{new}} = R + \boldsymbol{\phi}$: $\boldsymbol{\theta}^{i+1} = \boldsymbol{\theta}^i + \alpha \nabla_{\boldsymbol{\theta}^i} J^{\text{In}}(\boldsymbol{\theta}^i, \phi), i \in \{0, 1 \ldots K-1\}$. The meta-gradient estimation of the intrinsic reward matrix $\nabla_{\boldsymbol{\phi}} J^{\text{Out}}(\boldsymbol{\theta}^K)$ is needed in this case. Note that in the outer loss we use the original reward matrix R so the outer loss is $J^{\text{Out}}(\boldsymbol{\theta}^K)$ rather than $J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^K)$. Compared with MAML-RL, the object of meta-update (intrinsic matrix) and the object of inner-update (policy parameters) are different, which help us identify the problem mentioned in Sec. 4.2.

In this case, we choose the LVC and AD estimator. We conduct ablation study on inner-step and sample size shown in Fig. 1(c,d). With more sample size and less step size, the correlation increases for both estimator. Two important features are: (1) With 1-step inner-loop setting, both estimator performs similarly in the correlation. (2) With multi-step inner-loop setting, LVC based estimator can still reach relatively high correlation while MAML-biased estimator directly reaches low correlation after 5-step inner-loop. The phenomenon shown here corresponds exactly to the Hessian estimation issue we discuss in Sec. 4.2 and the bias issue will be more severe with multi-step inner-loop setting.

5.2 Compositional bias/off-policy learning in LOLA

In this subsection, we conduct three experiments on Iterated Prisoner Dilemma (IPD) with the LOLA algorithm to show: (1) The effect brought by different inner/outer estimators. (2) The effect brought compisitional bias (Sec. 4.1) (3) How off-policy correction (Sec. 4.4) can help the LOLA algorithm. Please refer to Appendix J.2.1, J.2.2 for more experimental setting and results.

Ablation on LOLA-DiCE inner/outer estimation. We report the result of conducting ablation study for different inner/outer-loop estimation of LOLA-DiCE in the Fig. 2(a). Here the inner-loop estimation refers to $\nabla_{\boldsymbol{\theta}} J^{\text{In}}(\boldsymbol{\phi}, \boldsymbol{\theta})$ while outer-loop estimation refers to $\nabla_{\boldsymbol{\theta}^1} J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^1)$ and $\nabla_{\boldsymbol{\phi}} J^{\text{Out}}(\boldsymbol{\phi}, \boldsymbol{\theta}^1)$. The return shown in Fig. 2(a) reveals us two findings: 1) The inner-loop gradient estimation plays an important role for making LOLA work—the default batch size 128 fails while the batch size 1024 succeeds. 2) The outer-loop gradient estimation is also crucial to the performance of LOLA-Exact. Furthermore, we continue conducting ablation studies on the inner-loop gradient update.

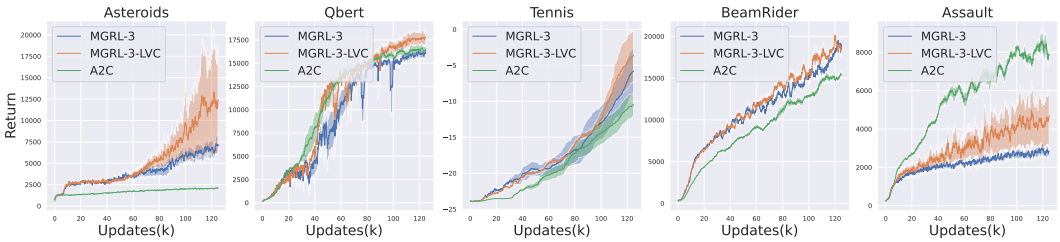

Figure 3: Experimental results on Atari game over 5 random seeds.

Ablation on compositional bias. Since the unbiased DiCE estimator is used in LOLA-DiCE algorithm, there is no Hessian estimation bias in the LOLA algorithm. Thus, we mainly discuss the problem of compositional bias brought in Fig. 2(b). We also apply the implementation in Sec. 5.1 to decompose meta-gradient estimation with different estimators. Fig. 2(b) show us the ablation study over compositional bias, which reveals that: compositional bias may decrease the performance and by adding more samples or using analytical solution, the performance can start to improve.

Off-policy DiCE and ablation study. We use the off-policy learning to conduct inner-loop update and keep the outer-loop gradient same as before. By combing DiCE and off-policy learning, we have off-policy DiCE $J^{\text{OFF-DICE}}$: $\mathbb{E}_{\tau}\left[\sum_{t=0}^{H-1}\left(\prod_{t'=0}^{t}\frac{\pi_{\phi}(\boldsymbol{a}_{t'}^{1}|\boldsymbol{s}_{t'}^{1})\pi_{\theta}(\boldsymbol{a}_{t'}^{2}|\boldsymbol{s}_{t'}^{2})}{\mu_{1}(\boldsymbol{a}_{t'}^{1}|\boldsymbol{s}_{t'}^{1})\mu_{2}(\boldsymbol{a}_{t'}^{2}|\boldsymbol{s}_{t'}^{2})}\right)R_t\right]$, where ϕ, θ refer to the current policy, μ_1, μ_2 refer the behaviour policy for agent 1 and agent 2, respectively. H is the trajectory length and R refers to the reward for agent. Note that the off-policy DiCE here can not only lower the compositional bias by lowering the first-order policy gradient error, but also helps lower the Hessian variance theoretically. By the decomposition trick, we conduct experiments by traversing over all learning settings for (off-off/off-on/on-off/on-on), which are shown at Fig. 2(d). Comparisons between different settings verify that off-policy DiCE can increase performance by either lowering the compositional bias, or the hessian variance, or both.

5.3 Multi-step Hessian correction on MGRL

Finally, we conduct experiment over MGRL [38]. See Appendix J.3.1 and J.3.2 for experimental settings. When applying the LVC estimator in MGRL, we get the new inner-loop update equation: $\mathbb{E}_{\tau\sim p(\tau;\theta)}\left[\sum_{t=0}^{H-1}\frac{\nabla_{\theta}\pi_{\theta}(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\perp\pi_{\theta}(\boldsymbol{a}_t|\boldsymbol{s}_t)}\left(g_{\boldsymbol{\phi}}(\boldsymbol{\tau})-v_{\theta}(\boldsymbol{s}_t)\right)\right]$, where $\boldsymbol{\phi} = (\gamma, \lambda)$ refer to meta-paramters and θ refers to the RL policy, $g_{\boldsymbol{\phi}}(\tau)$ denotes λ-return, $v_{\theta}(s_t)$ denotes value prediction. We conduct experiment on eight environments of Atari games. We follow previous work [4] to use the "discard" strategy in which we conduct multiple virtual inner-loop updates for meta gradient estimation. This strategy is designed to keep the inner learning update unchanged.

In Fig. (3), we show five environments comparing three variants of algorithm: 1) Baseline Advantage Actor-critic(A2C) algorithm [25]; 2) 3-step MGRL + A2C; 4) 3-step MGRL + A2C + LVC correction. The "3-step" means we take 3 inner-loop RL virtual updates for calculating meta-gradient. Refer to Appendix J.3.3 for experimental results on all eight environments. Compared with 3-step MGRL, the MGRL with LVC correction can substantially improves the performance, which validates the effectiveness of the multi-step Hessian correction in Sec. 4.4 for handling meta-gradient estimation bias and bring in better hyperparameter-tuning in RL. Note that the fact that A2C algorithms can achieve better results compared with 3-step MGRL is consistent with the results in [38].

6 Conclusion

In this paper, we introduce a unified framework for studying generic meta-gradient estimations in gradient-based Meta-RL. Based on this framework, we offer two theoretical insights that **1)** the compositional bias has an upper bound of $O\big(K\alpha^K\hat{\sigma}_{\text{In}}|\tau|^{-0.5}\big)$ with respect to the inner-loop update step K, the learning rate α, the estimate variance $\hat{\sigma}_{\text{In}}^2$ and the sample size $|\tau|$, and **2)** the multi-step Hessian bias $\hat{\Delta}_H$ has a polynomial impact of $O\big((K-1)(\hat{\Delta}_H)^{K-1}\big)$. To validate our theoretical discoveries, we conduct a comprehensive list of ablation studies. Empirical results over tabular MDP, LOLA-DiCE and MGRL validate our theories and the effectiveness of correction methods. We believe our work can inspire more future work on unbiased meta-gradient estimations in GMRL.

References

[1] Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. *arXiv preprint arXiv:1710.03641*, 2017.

[2] Atilim Gunes Baydin, Robert Cornish, David Martinez Rubio, Mark Schmidt, and Frank Wood. Online learning rate adaptation with hypergradient descent. *arXiv preprint arXiv:1703.04782*, 2017.

[3] Sarah Bechtle, Artem Molchanov, Yevgen Chebotar, Edward Grefenstette, Ludovic Righetti, Gaurav Sukhatme, and Franziska Meier. Meta learning via learned loss. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4161–4168. IEEE, 2021.

[4] Clément Bonnet, Paul Caron, Thomas Barrett, Ian Davies, and Alexandre Laterre. One step at a time: Pros and cons of multi-step meta-gradient reinforcement learning. *arXiv preprint arXiv:2111.00206*, 2021.

[5] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[6] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 1082–1092. PMLR, 2020.

[7] Alireza Fallah, Kristian Georgiev, Aryan Mokhtari, and Asuman Ozdaglar. On the convergence theory of debiased model-agnostic meta-reinforcement learning. *Advances in Neural Information Processing Systems*, 34:3096–3107, 2021.

[8] Gregory Farquhar, Shimon Whiteson, and Jakob Foerster. Loaded dice: trading off bias and variance in any-order score function estimators for reinforcement learning. *arXiv preprint arXiv:1909.10549*, 2019.

[9] Xidong Feng, Oliver Slumbers, Ziyu Wan, Bo Liu, Stephen McAleer, Ying Wen, Jun Wang, and Yaodong Yang. Neural auto-curricula in two-player zero-sum games. *Advances in Neural Information Processing Systems*, 34, 2021.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[11] Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 122–130, 2018.

[12] Jakob Foerster, Gregory Farquhar, Maruan Al-Shedivat, Tim Rocktäschel, Eric Xing, and Shimon Whiteson. Dice: The infinitely differentiable monte carlo estimator. In *International Conference on Machine Learning*, pages 1529–1538. PMLR, 2018.

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

[14] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In *International Conference on Machine Learning*, pages 1165–1173. PMLR, 2017.

[15] Haotian Fu, Hongyao Tang, Jianye Hao, Chen Chen, Xidong Feng, Dong Li, and Wulong Liu. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning. *arXiv preprint arXiv:2009.13891*, 2020.

[16] Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. *Advances in Neural Information Processing Systems*, 31:5302–5311, 2018.

[17] Yujing Hu, Weixun Wang, Hangtian Jia, Yixiang Wang, Yingfeng Chen, Jianye Hao, Feng Wu, and Changjie Fan. Learning to utilize shaping rewards: A new approach of reward shaping. *Advances in Neural Information Processing Systems*, 33, 2020.

[18] Kaiyi Ji, Junjie Yang, and Yingbin Liang. Multi-step model-agnostic meta-learning: Convergence and improved algorithms. *CoRR*, abs/2002.07836, 2020.

[19] Dong-Ki Kim, Miao Liu, Matthew Riemer, Chuangchuang Sun, Marwa Abdulhai, Golnaz Habibi, Sebastian Lopez-Cot, Gerald Tesauro, and Jonathan P How. A policy gradient algorithm for learning to learn in multiagent reinforcement learning. *arXiv preprint arXiv:2011.00382*, 2020.

[20] Louis Kirsch, Sjoerd van Steenkiste, and Juergen Schmidhuber. Improving generalization in meta reinforcement learning using learned objectives. In *International Conference on Learning Representations*, 2019.

[21] Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail, 2018.

[22] Alistair Letcher, Jakob Foerster, David Balduzzi, Tim Rocktäschel, and Shimon Whiteson. Stable opponent shaping in differentiable games. In *International Conference on Learning Representations*, 2018.

[23] Hao Liu, Richard Socher, and Caiming Xiong. Taming maml: Efficient unbiased meta-reinforcement learning. In *International Conference on Machine Learning*, pages 4061–4071. PMLR, 2019.

[24] Jingkai Mao, Jakob Foerster, Tim Rocktäschel, Maruan Al-Shedivat, Gregory Farquhar, and Shimon Whiteson. A baseline for any order gradient estimation in stochastic computation graphs. In *International Conference on Machine Learning*, pages 4343–4351. PMLR, 2019.

[25] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[26] Junhyuk Oh, Matteo Hessel, Wojciech M Czarnecki, Zhongwen Xu, Hado van Hasselt, Satinder Singh, and David Silver. Discovering reinforcement learning algorithms. *arXiv preprint arXiv:2007.08794*, 2020.

[27] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

[28] Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. Promp: Proximal meta-policy search. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[29] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[30] Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10): 1095–1100, 1953.

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

[32] Yunhao Tang, Tadashi Kozuno, Mark Rowland, Remi Munos, and Michal Valko. Unifying gradient estimators for meta-reinforcement learning via off-policy evaluation. *Advances in Neural Information Processing Systems*, 34, 2021.

[33] Vivek Veeriah, Matteo Hessel, Zhongwen Xu, Richard Lewis, Janarthanan Rajendran, Junhyuk Oh, Hado van Hasselt, David Silver, and Satinder Singh. Discovery of useful questions as auxiliary tasks. *arXiv preprint arXiv:1909.04607*, 2019.

[34] Vivek Veeriah, Tom Zahavy, Matteo Hessel, Zhongwen Xu, Junhyuk Oh, Iurii Kemaev, Hado van Hasselt, David Silver, and Satinder Singh. Discovery of options via meta-learned subgoals. *arXiv preprint arXiv:2102.06741*, 2021.

[35] Risto Vuorio, Jacob Austin Beck, Gregory Farquhar, Jakob Nicolaus Foerster, and Shimon Whiteson. No DICE: An investigation of the bias-variance tradeoff in meta-gradients. In *Deep RL Workshop NeurIPS 2021*, 2021.

[36] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

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

[38] Zhongwen Xu, Hado van Hasselt, and David Silver. Meta-gradient reinforcement learning. *arXiv preprint arXiv:1805.09801*, 2018.

[39] Zhongwen Xu, Hado van Hasselt, Matteo Hessel, Junhyuk Oh, Satinder Singh, and David Silver. Meta-gradient reinforcement learning with an objective discovered online. *arXiv preprint arXiv:2007.08433*, 2020.

[40] Tom Zahavy, Zhongwen Xu, Vivek Veeriah, Matteo Hessel, Junhyuk Oh, Hado van Hasselt, David Silver, and Satinder Singh. A self-tuning actor-critic algorithm. *arXiv preprint arXiv:2002.12928*, 2020.

[41] Zeyu Zheng, Junhyuk Oh, and Satinder Singh. On learning intrinsic rewards for policy gradient methods. *Advances in Neural Information Processing Systems*, 31:4644–4654, 2018.

[42] Zeyu Zheng, Junhyuk Oh, Matteo Hessel, Zhongwen Xu, Manuel Kroiss, Hado Van Hasselt, David Silver, and Satinder Singh. What can learned intrinsic rewards capture? In *International Conference on Machine Learning*, pages 11436–11446. PMLR, 2020.

[43] Wei Zhou, Yiying Li, Yongxin Yang, Huaimin Wang, and Timothy Hospedales. Online meta-critic learning for off-policy actor-critic methods. *Advances in Neural Information Processing Systems*, 33, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Appendix B.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sec. 4
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix G and Appendix H
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] , See Appendix J

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] , See Fig. 1, 2, 3.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix J

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes]

   (b) Did you mention the license of the assets? [Yes]

   (c) Did you include any new assets either in the supplemental material or as a URL? [No]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]