Taming MAML: Efficient Unbiased Meta-Reinforcement Learning

Hao Liu [1] **Richard Socher** [1] **Caiming Xiong** [1]

Abstract

While meta reinforcement learning (Meta-RL) methods have achieved remarkable success, obtaining correct and low variance estimates for policy gradients remains a significant challenge. In particular, estimating a large Hessian, poor sample efficiency and unstable training continue to make Meta-RL difficult. We propose a surrogate objective function named, Taming MAML (TMAML), that adds control variates into gradient estimation via automatic differentiation. TMAML improves the quality of gradient estimation by reducing variance without introducing bias. We further propose a version of our method that extends the meta-learning framework to learning the control variates themselves, enabling efficient and scalable learning from a distribution of MDPs. We empirically compare our approach with MAML and other variance-bias trade-off methods including DICE, LVC, and action-dependent control variates. Our approach is easy to implement and outperforms existing methods in terms of the variance and accuracy of gradient estimation, ultimately yielding higher performance across a variety of challenging Meta-RL environments.

1. Introduction

Recent progress in deep reinforcement learning (Deep RL) has achieved very impressive results in domains ranging from playing games (Mnih et al., 2015; Silver et al., 2016; 2017; Vinyals et al., 2019), to applications in program synthesis (Liang et al., 2018) and robotics (Andrychowicz et al., 2018; Gu et al., 2017; Liu et al., 2019). Despite this progress, Deep RL suffers from high sample complexity in learning even a single task and often fails to generalize to new situations. In contrast, human intelligence is capable of adapting to new situations in the face of limited experience. Meta reinforcement learning (Meta-RL) (Wang et al., 2016; Duan et al., 2016; Mishra et al., 2018; Finn et al., 2017; Nichol et al., 2018) aims to mitigate this issue by acquiring inductive bias in a data-driven manner. Specifically, Meta-RL considers a distribution of different yet related tasks that differ in, for instance, the reward function or the transition probabilities from one state to another. Its objective is to enable artificial agents to efficiently learn new tasks by learning strategies for fast adaptation from prior tasks (a.k.a, learning to learn) (Schmidhuber, 1987; Thrun & Pratt, 2012). Recent work demonstrates this approach to be promising in a diverse range of reinforcement learning tasks including robotics, synthetic rewards, and model-based learning, among others (Clavera et al., 2018; Xu et al., 2018b).

One particular type of gradient-based Meta-RL is Model Agnostic Meta Learning (MAML) (Finn et al., 2017), which directly trains for model parameters that can quickly adapt to individual tasks with standard gradient descent. This method allows acceleration of learning to achieve similar asymptotic performance as learning from scratch on new tasks. Despite its wide utilization, Meta-RL, in particular gradient-based Meta-RL, suffers from biased and high variance estimates for policy gradients. This is due to the small number of samples and the difficulty of calculating the Hessian, leading to poor sample efficiency and unstable behavior during training. Prior work in gradient-based Meta-RL mostly used biased gradients, either due to biased estimation in implementation (Finn et al., 2017), trading off bias for lower variance estimation (Rothfuss et al., 2019), or neglecting the complicated dependencies (Al-Shedivat et al., 2018; Stadie et al., 2018; Nichol et al., 2018). An existing method for unbiased estimation of Meta-RL gradients (Foerster et al., 2018) suffers from high variance and doesn't work well in challenging, continuous control tasks.

To tackle this problem, we focus on the control variate method, one of the most widely used variance reduction techniques in policy gradient. The idea of the control variate method is to subtract a Monte Carlo gradient estimator by a baseline function that analytically has zero expectation. If the baseline function is properly chosen such that it cancels out the variance of the original gradient estimator, it can achieve a lower variance estimation without introducing bias. Different variance reduction methods stem from different control variates. For example, in REINFORCE (Williams,

[1]Salesforce Research, Palo Alto, USA. Correspondence to: Hao Liu <lhao499@gmail.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

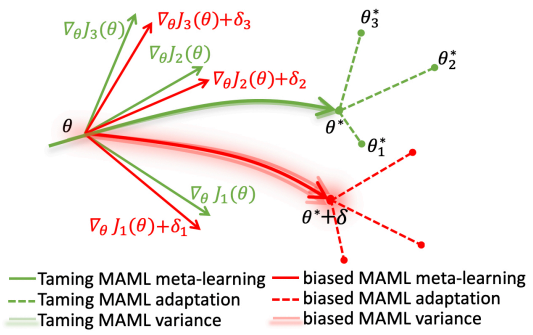

Figure 1. Diagram comparing MAML and our method, TMAML. A biased implementation of MAML (e.g, Finn et al. (2017); Rothfuss et al. (2019)) is illustrated in red. TMAML is illustrated in green. Meta-learning steps are shown using solid lines while adaptation is shown in dashed lines. δ refers to bias while shading refers to variance. In contrast to MAML, TMAML provides unbiased and low variance gradient estimates to optimize θ to quickly adapt to new tasks.

1992), a constant baseline function is chosen as a control variate; advantage actor-critic (A2C) (Sutton & Barto, 2018) considers a state-dependent baseline function as the control variate, which is often set to be an estimated value function.

Despite the effectiveness of control variates, incorporating them into the gradient of Meta-RL turns out to be fairly challenging due to the involved calculation of the Hessian and the gradient. In this work, we propose a simple and novel surrogate objective that incorporates control variates into gradient estimation in the context of Meta-RL, allowing low variance and unbiased gradient estimations, as shown in Figure 1. In addition, we introduce a method to meta-learn the control variates themselves, enabling efficient and scalable learning from a distribution of MDPs. We call this technique Taming MAML (TMAML). TMAML can be convenient implemented via automatic differentiation toolboxes and is well suited to combine and improve other gradient-based Meta-RL methods. We provide an analytical motivation for TMAML and demonstrate experimentally that it achieves substantial variance reduction during gradient estimation. Furthermore, we show that TMAML improves performance compared to existing Meta-RL methods on several challenging benchmarks.

Finally, we evaluate TMAML and meta-learning trained control variates on an extensive set of environments. We compare against recent approaches including DICE (Foerster et al., 2018) and LVC (Rothfuss et al., 2019), based on recent optimization method Proximal Policy Optimization (PPO) (Schulman et al., 2017), and demonstrate the effectiveness of TMAML in both variance reduction and sample efficiency.

2. Related Work

Learning agents that are capable of doing well in a distribution of different but related tasks is a long standing challenge. Perhaps most relevant are works related to the concept of *meta-learning* or *learning to learn* (Schmidhuber, 1987; Thrun & Pratt, 2012) which aims to discover, from experience, in order to accelerate learning process in unseen problem settings. Different approaches have been investigated in the literature. One type of approach trains recurrent networks on a set of environments/datasets drawn i.i.d. from the distribution to exploit the structure of multiple tasks to output the parameters of the trained model (Chen et al., 2017; Andrychowicz et al., 2016; Ravi & Larochelle, 2016) or directly output the prediction on given inputs (Duan et al., 2016; Mishra et al., 2018; Santoro et al., 2016; Xu et al., 2018a). Alternatively, the parameters of the model can be explicitly trained in such a way that they can learn a new task from the task distribution in only a small number of gradient steps (Finn et al., 2017; Nichol et al., 2018). Despite the wide utilization of this gradient-based approach, it suffers from biased or high variance gradient estimate.

Our work, which builds on the latter class of approaches, is also related to recent work on variance reduction for policy gradient using control variates in various settings (Grathwohl et al., 2018; Wu et al., 2018; Liu et al., 2017; Mao et al., 2019a). Tucker et al. (2018) compare recent different variance reduction method analytically. Learning task-dependent policies or value functions for multiple goals within similar environments was proposed in Schaul et al. (2015). The issue of bias-variance trade-off in policy gradient estimation has also received previous attention (Heess et al., 2015). Similarly, Xu et al. (2018b) learns to find meta-parameters (e.g., discount factor) such that the agent can achieve bias-variance trade-off to learn efficiently within a single task, while Schulman et al. (2016) analyse different effects of meta-parameters in return. Variance reduction for general stochastic computational graphs is discussed in Schulman et al. (2015) and Weber et al. (2019). In parallel to our work, Mao et al. (2019b) proposes using control variates to reduce variance of high order gradient estimation, our work differs in proposing a practical algorithm and meta-learn scalable meta control variates and empirically demonstrate its effectiveness in a set of Meta-RL tasks.

3. Background

This section establishes the notation and provides an overview of policy gradient and gradient-based Meta-RL.

3.1. Reinforcement Learning and Policy Gradient

We consider a discrete-time Markov decision process (MDP), defined by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \rho_0, r, \gamma)$, where $\mathcal{S} \subseteq \mathbb{R}^n$ is

a set of n-dimensional states, $\mathcal{A} \subseteq \mathbb{R}^m$ is a set of m-dimensional actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition probability distribution, $\rho_0 : \mathcal{S} \to [0, 1]$ is the distribution over initial states, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1]$ is the discount factor. We denote a stochastic policy as $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, which aims to optimize the expected return $\eta(\pi) = \mathbb{E}_\tau \left[\sum_{t=0}^{\infty} r(s_t, a_t)\right]$, where $\tau = (s_0, a_0, ...)$ is the trajectory following $s_0 \sim \rho_0$, $a_t \sim \pi(a_t|s_t)$, $s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)$. We use $V_\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, a_{t+1}, ...} \left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l}, a_{t+l})\right]$ to define the value function, and $Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, ...} \left[\sum_{l=0}^{\infty} \gamma^l r(s_{t+l}, a_{t+l})\right]$ to define the state-action value function.

Policy gradient methods estimate the gradient of expected returns with respect to the policy parameters (Sutton & Barto, 2018). To train a policy π_θ parameterized by θ, the Policy Gradient Theorem (Sutton & Barto, 2018) states that

$$\nabla_\theta \eta(\pi_\theta) = \mathbb{E}_{\substack{s \sim \rho_\pi \\ a \sim \pi_\theta}} \left[\nabla_\theta \log \pi_\theta(a|s) Q_{\pi_\theta}(s, a)\right], \quad (1)$$

where $\rho_\pi(s) = \sum_{t=0}^{\infty} \left[\gamma^t \Pr(s_t = s)\right]$ denotes the discounted state visitation frequency. Practical algorithms often use the undiscounted state visitation frequency (i.e., $\gamma = 1$ in ρ_π), which can make the estimation slightly biased (Thomas, 2014).

Estimating the policy gradient using Monte Carlo estimation for the Q function suffers from high variance (Mnih et al., 2016; Liu et al., 2017; Grathwohl et al., 2018; Wu et al., 2018). To reduce variance, an appropriately chosen baseline $b(s_t)$ can be subtracted from the Q-estimate without introducing bias (Greensmith et al., 2004). The policy gradient estimation with a baseline in Eq. 1 becomes $\mathbb{E}_{\rho_\pi, \pi_\theta} \left[\nabla_\theta \log \pi_\theta(a|s) \left(Q_{\pi_\theta}(s, a) - b(s)\right)\right]$. While an optimal baseline (Greensmith et al., 2004; Wu et al., 2018) and action-dependent baseline exist (Liu et al., 2017; Grathwohl et al., 2018), they are hard to estimate and are often replaced by the value function $b(s_t) = V_\pi(s_t)$ (Sutton & Barto, 2018).

Proximal Policy Optimization (PPO) (Schulman et al., 2017; Heess et al., 2017) is recent method for policy optimization. It uses a proximal KL divergence penalty to regularize and stabilize the policy gradient update. Given an existing policy π_{old}, PPO obtains a new policy by maximizing the following surrogate loss function

$$J_{\text{ppo}}(\theta) = \mathbb{E}_{\pi_{\text{old}}} \left[\frac{\pi_\theta(a|s)}{\pi_{\text{old}}(a|s)} Q_\pi(s, a) - \lambda \text{KL}\left[\pi_{\text{old}}(\cdot|s) \;||\; \pi_\theta(\cdot|s)\right]\right]$$

where the first term is an approximation of the expected reward, and the second term enforces the the updated policy to be close to the previous policy under KL divergence.

Using a baseline in the TRPO objective, i.e. replacing $Q_\pi(s, a)$ with $Q_\pi(s, a) - b(s)$, empirically improves policy performance (Schulman et al., 2016).

3.2. Gradient-based Meta-reinforcement learning

Deep RL typically requires an enormous number of samples to learn a single task and rarely allows generalization to new tasks, even similar ones. Meta-RL aims to use a distribution of tasks $\rho(\mathcal{T})$ to learn a policy that can be rapidly adapted to optimize performance on a specific task $\mathcal{T}$ from the distribution, where each task corresponds to a distinct MDP. Tasks are generally defined such that they share a state and action space but differ somehow with regard to transition dynamics and/or reward function.

Policy gradient in the Meta-RL context therefore equates to learning the parameters θ of a policy π_θ such that the parameters can be optimized for a specific task $\mathcal{T} \sim \rho(\mathcal{T})$ with only one or several vanilla policy gradient (VPG) updates. This is the motivation behind MAML (Finn et al., 2017), which aims to maximize the following objective:

$$J(\theta) = \mathbb{E}_{\mathcal{T} \sim \rho(\mathcal{T})} \left[\mathbb{E}_{\boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')} \left[R(\boldsymbol{\tau}')\right]\right] \quad (2)$$
$$\text{with } \theta' := U(\theta, \mathcal{T}) = \theta + \alpha \nabla_\theta \mathbb{E}_{\boldsymbol{\tau} \sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)} \left[R(\boldsymbol{\tau})\right],$$

where $\mathcal{T} \sim \rho(\mathcal{T})$ denotes the task sampled from task distribution, $R(\tau)$ denotes the sum of rewards along a trajectory $\boldsymbol{\tau} = \{(s_t, a_t, r_t)\}_{t=0}^N$, $\rho(\mathcal{T})$ and U denotes the task-dependent update function and is equivalent to performing one VPG step towards maximizing the performance of the policy on task $\mathcal{T}$, $P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)$ denotes probability of sampled trajectory $\boldsymbol{\tau} = \{(s_t, a_t, r_t)\}_{t=0}^N$ with policy θ and task $\mathcal{T}$. It is worth noting that this concept could be extended to handle multiple update steps, though we only consider the single step version here.

4. Method

In this section we first analyze challenges in obtaining unbiased and low variance estimates of gradients in MAML (Finn et al., 2017) and then briefly recap recent two recent advances: DICE (Foerster et al., 2018) and LVC (Rothfuss et al., 2019). Finally, we introduce our method, called TMAML, which is a surrogate function used to incorporate control variates into gradient estimates and meta-learned control variates.

Obtaining unbiased and low variance estimates of gradients in Meta-RL proves challenging due to the difficulty of estimating the Hessian, As discussed by Foerster et al. (2018), the score function surrogate objective approach is ill suited for calculating higher order derivatives via automatic differentiation toolboxes. This important fact was overlooked in the original RL-MAML implementation (Finn et al., 2017) leading to incorrect meta-gradient estimates which do not comply with the theory. We can write the gradient of the

meta-learning objective as

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\Big[\mathbb{E}_{\boldsymbol{\tau}'\sim P_\mathcal{T}(\boldsymbol{\tau}'|\theta')}\big[\nabla_{\theta'}\log P_\mathcal{T}(\boldsymbol{\tau}'|\theta') \\ R(\boldsymbol{\tau}')\nabla_\theta U(\theta,\mathcal{T})\big]\Big], \tag{3}$$

Since the update function U resembles a policy gradient step, its gradient $\nabla_\theta\theta' = \nabla_\theta U(\theta,\mathcal{T})$ involves computing the Hessian of the reinforcement learning objective, i.e., $\nabla^2_\theta\,\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right]$.

This can be shown as:

$$\begin{aligned}
\nabla_\theta\theta' &= \nabla_\theta\left(\theta + \alpha\,\nabla_\theta\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right]\right),\ \alpha \text{ is a step size} \\
&= I + \alpha\nabla^2_\theta\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right] \\
&= I + \alpha\nabla_\theta\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\nabla_\theta\log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau})\right] \\
&= I + \alpha\nabla_\theta\int P_\mathcal{T}(\boldsymbol{\tau}|\theta)\nabla_\theta\log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau})d\boldsymbol{\tau} \\
&= I + \alpha\int P_\mathcal{T}(\boldsymbol{\tau}|\theta)\nabla^2_\theta\log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau})d\boldsymbol{\tau} \\
&\quad + \alpha\int P_\mathcal{T}(\boldsymbol{\tau}|\theta)\nabla_\theta\log\pi_\theta(\boldsymbol{\tau})\nabla_\theta\log\pi_\theta(\boldsymbol{\tau})^\top R(\boldsymbol{\tau})d\boldsymbol{\tau}
\end{aligned}$$

Due to the Hessian involved in the gradient estimation, even when properly implemented to obtain an unbiased estimation, we show that the meta-gradients exhibit high variance. Specifically, the estimation of the Hessian in the RL-objective, which is inherent in the meta-gradients, requires special consideration. In MAML (Finn et al., 2017), the Hessian is ignored during implementation, the $\nabla_\theta\theta'$ is approximated by $I + \int P_\mathcal{T}(\boldsymbol{\tau}|\theta)\nabla^2_\theta\log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau})d\boldsymbol{\tau}$, giving a biased gradient estimate.

In this section, we motivate and introduce an unbiased and low variance estimate: an improved estimator for the Hessian of the RL-objective by using a control variate, trained using meta-learning, which promotes better meta-policy gradient updates.

Thus, the problem is how to estimate

$$\nabla^2_\theta J_{\text{inner}} = \nabla^2_\theta\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right] \tag{4}$$

We quickly recall the main results from Foerster et al. (2018) which aims to provide an unbiased estimate of $\nabla^2_\theta J_{\text{inner}}$ but still suffers from high variance. Then, we recap a recent work on variance-bias trade-off (Rothfuss et al., 2019) to get a biased but low variance estimate of $\nabla^2_\theta J_{\text{inner}}$.

Unbiased estimation In DICE (Foerster et al., 2018), the surrogate function is given by:

$$J^{\text{DICE}} = \sum_{t=0}^{H-1}\left(\prod_{t'=0}^{t}\frac{\pi_\theta(a_{t'}|s_{t'})}{\perp(\pi_\theta(a_{t'}|s_{t'}))}\right)r(s_t,a_t), \tag{5}$$

where $\perp$ denotes the 'stop gradient' or 'detach' operation in automatic differentiation framework. In expectation, the Monte Carlo estimate of the Hessian $\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}[\nabla^2_\theta J^{\text{DICE}}]$ is equivalent to the Hessian of the inner objective in Equation (4):

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}[\nabla^2_\theta J^{\text{DICE}}] \\
&= \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\Bigg[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\nabla_\theta\log\pi_\theta(a_{t'}|s_{t'})\right) \\
&\quad\left(\sum_{t'=0}^{t}\nabla_\theta\log\pi_\theta(a_{t'}|s_{t'})\right)^\top r(s_t,a_t) + \\
&\quad\left(\sum_{t'=0}^{t}\nabla^2_\theta\log\pi_\theta(a_{t'}|s_{t'})\right)r(s_t,a_t)\Bigg] \\
&= \nabla^2_\theta J_{\text{inner}}
\end{aligned}$$

however, DICE unbiased estimation $\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}[\nabla^2_\theta J^{\text{DICE}}]$ suffers from high variance.

Biased, lower variance estimation Rothfuss et al. (2019) propose a surrogate function called LVC which is biased and lower variance,

$$J^{\text{LVC}} = \sum_{t=0}^{H-1}\frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))}\left(\sum_{t'=t}^{H-1}r(s_{t'},a_{t'})\right) \tag{6}$$

In expectation, $\nabla^2_\theta J^{\text{LVC}}$ is approximately equivalent to $\nabla^2_\theta J_{\text{inner}}$:

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\nabla^2_\theta J^{\text{LVC}}\right] = \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)} \\
&\quad\Bigg[\sum_{t=0}^{H-1}\Big(\sum_{t'=0}^{t}\nabla_\theta\log\pi_\theta(\boldsymbol{a}_{t'}|\boldsymbol{s}_{t'})\nabla_\theta\log\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)^\top\Big) \\
&\quad r(\boldsymbol{s}_t,\boldsymbol{a}_t)\Bigg] + \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\Bigg[\sum_{t=0}^{H-1} \\
&\quad\Big(\sum_{t'=0}^{t}\nabla^2_\theta\log\pi_\theta(\boldsymbol{a}_{t'}|\boldsymbol{s}_{t'})\Big)r(\boldsymbol{s}_t,\boldsymbol{a}_t)\Bigg] \\
&\quad\approx \nabla^2_\theta J_{\text{inner}}
\end{aligned}$$

We leave the derivation of DICE in Section A and the derivation of LVC in Section B in the supplementary file. Note that in J^{LVC}, the bias term is not quantified, which can be harmful for gradient optimization.

4.1. Meta-RL for a distribution of MDPs

We now formally define a distribution of task-dependent MDPs and derive variance-reducing control variates for policy gradient methods.

In this setting, the environment of the agent is a distribution of different yet related tasks that differ in, for instance, the

reward function or in the probabilities of transitions from one state to another.

Definition 1. *A distribution of task-dependent MDPs is defined by $(\mathcal{S}, \mathcal{A}, \mathcal{P}_s, \mathcal{P}_\mathcal{T}, \rho_0^s, \rho_0^z, r, \gamma)$, $\mathcal{P}_s(s_{t+1}|s_t, a_t, \mathcal{T}_i)$ is the transition kernel of the states for i-th MDP in the group, $\mathcal{P}_\mathcal{T}(\mathcal{T}_{i+1}|\mathcal{T}_i)$ is the transition kernel of the input process, $\rho^\mathcal{T}(\mathcal{T})$ is the distribution of tasks, $r(s_t, a_t, \mathcal{T}_i)$ is the reward function, and $\mathcal{S}$, $\mathcal{A}$, ρ_0^s, γ follow the standard definition in reinforcement learning in Section 3.1.*

A task-dependent MDP adds a task description, $\mathcal{T}_i$, to a standard MDP. In a task-dependent MDP, the next state s_{t+1} depends on $(s_t, a_t, \mathcal{T}_i)$. We seek to learn policies that maximize cumulative expected rewards on such a group of MDPs. We now consider policy gradient methods for learning a policy for a group of task-dependent MDPs.

4.2. Surrogate function to incorporate control variates

In this section, we propose a surrogate function, which when twice differentiated, can incorporate a control variate for each task.

In order to obtain unbiased and low variance estimates, we propose to incorporate a control variate into gradient estimation

$$J_i^{\text{TMAML}} = \sum_{t=0}^{H-1} \left[1 - \left(\prod_{t'=0}^{t} \frac{\pi_\theta(a_{t'}|s_{t'})}{\perp(\pi_\theta(a_{t'}|s_{t'}))}\right)\right] \left(1 - \frac{\pi_\theta(a_t|s_t, z)}{\perp(\pi_\theta(a_t|s_t, z))}\right) b(s_t, \mathcal{T}_i) \tag{7}$$

where i represents the i-th task, and $b(s_t, \mathcal{T}_i)$ is a baseline function that depends on state s_t and task $\mathcal{T}_i$. Here, $b(s_t, \mathcal{T}_i)$ serves as the control variate. Generalization of the above to depend on action is also possible, which we will show below.

Theorem 1. *With a proper control variate $b(s_t, \mathcal{T}_i)$, we can derive*

$$\mathbb{E}_{\boldsymbol{\tau} \sim P_{\mathcal{T}_i}(\boldsymbol{\tau}|\theta)}[\nabla_\theta^2 J_i^{TMAML}] = 0 \tag{8}$$

Proof. We leave the proof to Section C in the supplementary file. □

Theorem 1 means we can combine J_i^{TMAML} with existing gradient estimates such as J_i^{DICE} without introducing any bias. That is, for each task $\mathcal{T}_i$:

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{\tau} \sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)} \left[\nabla_\theta^2 \left(J_i^{\text{DICE}} + J_i^{\text{TMAML}}\right)\right] \\ &= \mathbb{E}_{\boldsymbol{\tau} \sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)} \left[\nabla_\theta^2 J_i^{\text{DICE}}\right] \\ &= \nabla_\theta^2 J_{\text{inner}} \end{aligned} \tag{9}$$

Theorem 2. *J_i^{TMAML} is a surrogate function to incorporate control variates $b(s_t, \mathcal{T}_i)$ into gradient estimation via automatic differentiation toolboxes efficiently to reduce gradient variance without introducing any bias into gradient estimation, such as when combined with J_i^{DICE}.*

Proof. The detailed proof is left in Section C in supplementary file. To give an intuition, taking the Hessian of J_i^{TMAML} combined with J_i^{DICE} gives

$$\begin{aligned} &\nabla_\theta^2 \left(J_i^{\text{DICE}} + J_i^{\text{TMAML}}\right) \rightarrow \\ &\sum_{t=0}^{H-1} \left[\nabla_\theta^2 \log \pi_\theta(a_t|s_t, z) \left(\sum_{t'=0}^{t} r(a_{t'}, s_{t'}, z)\right)\right] \\ &+ 2 \sum_{t=0}^{H-1} \left[\nabla \log \pi_\theta(a_t|s_t, z)^\top \left(\sum_{t'=t}^{H-1} \nabla \log \pi_\theta(a_{t'}|s_{t'}, z) \right.\right. \\ &\left.\left. \left(\sum_{k=t'}^{H-1} r(a_k, s_k) - b(s_t, \mathcal{T}_i)\right)\right)\right], \end{aligned} \tag{10}$$

where $\rightarrow$ denotes evaluate value.

As long as $b(s_t, \mathcal{T}_i)$ is a good baseline function that correlates with $\sum_{k=t'}^{H-1} r(a_k, s_k)$, then the product $\left(\sum_{t'=t}^{H-1} \nabla \log \pi_\theta(a_{t'}|s_{t'}, z) \left(\sum_{k=t'}^{H-1} r(a_k, s_k) - b(s_t, \mathcal{T}_i)\right)\right)$ has a lower variance than the same without $b(s_t, \mathcal{T}_i)$, leading to a low variance estimate of $\nabla_\theta^2 \left(J_i^{\text{DICE}} + J_i^{\text{TMAML}}\right)$. □

Theorem 2 states that by auto-differentiating twice, our surrogate function can incorporate control variates into the gradient estimate, which can reduce the variance of estimating the Hessian.

Given the unbiased estimates, the next question is how to learn an effective control variate to reduce variance. We now present two approaches that exploit the distribution of related MDPs to learn task-dependent control variates efficiently.

Per-task control variates A straightforward way to learn control variates is to have a separate control variates for each task $\mathcal{T}_i$. We use the recently proposed action-dependent control variates (Wu et al., 2018) and extend it to task dependent reinforcement learning. It is straightforward to show that the optimally of action-dependent control variates still holds for each task $\mathcal{T}_i$ since we can view task id $\mathcal{T}_i$ as part of state s_t. Although it maybe not optimal control varaites for reducing variance of $\nabla_\theta J(\theta)$. Specifically, for each task $\mathcal{T}_i$, we learn the control variate for the current task as

$$\begin{aligned} &b_i\left(s_t, a_t^{-i}, \mathcal{T}_i\right) \\ &= \mathbb{E}_{a_j \sim \pi(a_t^i|s_t)} \left[Q_{\pi_\theta}\left(s_t, \left(a_t^{-i}, \alpha_j, z\right)\right)\right], \end{aligned} \tag{11}$$

where $a_j \sim \pi(a_t^i|s_t)$ means a_j sampled from action coordinate a_i. For each $\mathcal{T}_i$, we learn an optimal action-dependent

control variate. The per-task optimal control variates has two disadvantages: first, it does not scale if the task distribution contains a large number of task; second, it does not exploit the similarity between different yet related environments. In order to utilize samples from other tasks, we derive a meta control variate for each task which utilizes samples from other tasks.

Algorithm 1 Learning Meta Control Variates

Require: Step size α', β'; meta control variates parameters θ_V; episodes $\mathcal{D} = \{\tau_{1:N}\}$.

repeat
  Adapt θ_V with the first half of episodes $\{\tau_{1:N/2}\}$: $\theta_V^1 = \theta_V - \alpha' \nabla_{\theta_V} \mathcal{L}_{\theta_V}(\tau_{1:N/2})$
  Estimate meta control variates $V_{\theta_V^1}(s_t)$ for $s_t \sim \tau_{N/2:N}$ using adapted θ_V^1 with Equation (12)
  Adapt θ_V with the second half of episodes $\{\tau_{N/2:N}\}$: $\theta_V^2 = \theta_V - \alpha \nabla_{\theta_V} \mathcal{L}_{\theta_V}(\tau_{N/2:N})$
  Estimate meta control variates $V_{\theta_V^2}(s_t)$ for $s_t \sim \tau_{1:N/2}$ using adapted θ_V^2 with Equation (12)
  Update meta control variates: $\theta_V \leftarrow \theta_V - \beta' \nabla_{\theta_V} \mathcal{L}_{\theta_V^1}(\tau_{N/2:N}) - \beta' \nabla_{\theta_V} \mathcal{L}_{\theta_V^2}(\tau_{1:N/2})$
  Swap trajectories $\{\tau_{1:N/2}\}$ and $\{\tau_{N/2:N}\}$
until θ_V converge

Meta control variates Ideally, we would like an approach that enables shared learning across different tasks. We present a method based on meta learning to maximize the use of information across tasks. The idea is to use all tasks to learn a meta baseline function. The part that learns the weights of this meta baseline uses MAML.

Note that here we use the original implementation to learn meta control variate, which means the resulting control variates may be not exactly the one we want. Fortunately, it can be proved that the gradient still remains unbiased as long as the control variates depends on state and task (see Theorem 1 for details).

The pseudocode in Algorithm 1 describes the training algorithm for learning control variates. We follow the notation of MAML, denoting the loss in the value function $V_{\theta_V}(\cdot)$ as

$$\mathcal{L}_{\theta_V}(\tau_{i:j}) = \sum_{s_t, a_t, r_t \sim \tau_{i:j}} \| V_{\theta_V}(s_t) - \sum_{t'=t}^{T} \gamma^{t'-t} r_t \|^2, \quad (12)$$

We use the first half trajectories to do adaptation of meta control variates and then compute inner loss of meta control variates on the other half of trajectories, then we do outer gradient update using inner loss as in MAML. Finally, we swap the two groups of trajectories and repeat the same process until training loss converges. We do not use the same rollouts to adapt the meta control variates and compute the inner loss of it to avoid introducing extra bias to the baseline function (Tucker et al., 2018). The pseudocode in Algorithm 2 combines TMAML and meta control variates together, and describes the full training algorithm for optimizing θ to quickly adapt to new tasks.

Algorithm 2 Meta-RL with TMAML Gradient Estimators

Require: Task distribution ρ, step sizes α, β.

Randomly initialize policy parameters θ and meta control variates parameters θ_V.
repeat
  Sample batch of tasks $\mathcal{T}_i \sim \rho(\mathcal{T})$
  // Fitting control variates
  Learn meta control variates with Algorithm 1
  (or learn per task control variates for each $\mathcal{T}_i$ with Equation (11))
  for all $\mathcal{T}_i$ **do**
    // Inner gradient update θ
    Sample pre-update episodes $\mathcal{D}_i = \{\tau_i\}$ from $\mathcal{T}_i$ using π_θ
    Compute adapted parameters $\theta'_{\mathcal{T}_i} \leftarrow \theta + \alpha \nabla_\theta (J_{\mathcal{T}_i}^{\text{TMAML}}(\theta) + J_{\mathcal{T}_i}^{\text{DICE}}(\theta))$ with $\mathcal{D}_i = \{\tau_i\}$
    Sample post-update episodes $\mathcal{D'}_i = \{\tau'_i\}$ from $\mathcal{T}_i$ using $\pi_{\theta'}$
  end for
  // Outer gradient update θ
  Update $\theta \leftarrow \theta + \beta \sum_{\mathcal{T}_i} \nabla_\theta (J_{\mathcal{T}_i}^{\text{TMAML}}(\theta'_{\mathcal{T}_i}) + J_{\mathcal{T}_i}^{\text{DICE}}(\theta'_{\mathcal{T}_i}))$ using each $\mathcal{D}'_i = \{\tau'_i\}$
until θ converge

5. Experiment

5.1. Task Descriptions

In order to evaluate the effectiveness of our algorithm, we examine its performance across several standardized continuous control environments as implemented in the OpenAI Gym (Brockman et al., 2016) in the MuJoCo physics simulator (Todorov et al., 2012).

We list environments used in this paper here. Each environment contains a distribution of tasks for the agent to achieve. Hyperparameters used in each environment can be found in supplementary file.

Ant reaching target coordinates. In this environment, each task is associated with a location randomly chosen from a circle in the XY plane that the agent must reach. The goal location is not given to the agent. Instead, the agent must learn to locate, approach, and stop at the target. The agent receives at each time step a reward composed of a control cost, a contact cost, a survival reward, and a penalty equal to its L1 distance to the target position. The tasks are generated by sampling the target positions from the uniform distribution on $[-3, 3]^2$.

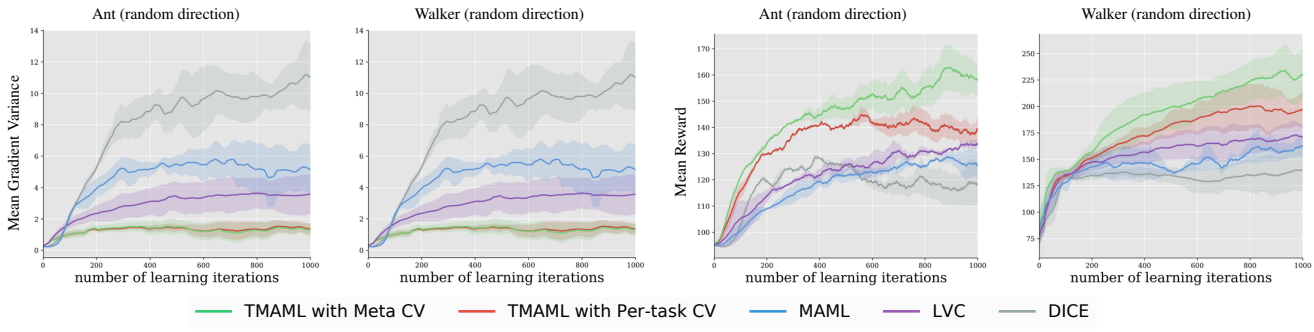

Figure 2. Variance reduction and performance improvements with TMAML. (Left) Average variance of gradients sampled across different mini-batches during evaluation. (Right) Mean task reward during evaluation. For consistency with published techniques, we use the TRPO algorithm for everything except LVC, which is intended to make use of PPO. Shaded area represents one standard deviation, learning curves are on 5 random seeds.

Ant, Humanoid, Walker, HalfCheetah moving forward or backward. The task is chosen between two directions: forward and backward. The agent must run along the goal direction as far as possible and receives at each time step a reward composed of a control cost and a reward equal to its velocity in the target direction. The tasks are generated by sampling the target directions from a Bernoulli distribution on $\{-1, 1\}$ with parameter 0.5 (-1: backward, +1: forward).

Ant and Humanoid reaching random direction in XY plane. Each task corresponds to a random direction in the XY plane. The agent receives a reward composed of a control cost and a penalty equal to the difference between its current direction and the target direction.

Walker, Swimmer, and HalfCheetah reaching random velocity. Each task corresponds to a random velocity, and the agent receives at each time step a reward composed of a control cost and a penalty equal to the difference between its current velocity and the target velocity. The tasks are generated by sampling the target velocities from the uniform distribution on $[0, 2]$.

5.2. Variance of Gradient Estimation

We start by attempting to empirically validate that TMAML indeed reduces the variance of the gradient estimates without introducing bias or detracting from performance. To that end, we compare both TMAML variants with MAML, LVC, and DICE on two tasks where we exhaustively resample gradients after each learning iteration. We compare the gradients estimated from each such sample to measure their variance. Figure 2 (left) shows the results, which clearly show that gradients estimated with TMAML are the most consistent across samples. Importantly, this is unlikely to stem from some trivial degeneracy within the gradients, as TMAML also yields the highest average reward among the examined methods, see Figure 2 (right). These empirical results support our analytical conclusion that TMAML reduces variance without introducing bias. While both TMAML variants substantially reduce variance, meta-learned control variates seems to perform better.

5.3. Accuracy of Gradient Estimation

To further confirm that TMAML yields high quality gradients, we train a small network on a toy task where we can estimate the ground-truth gradient (because calculating the Hessian becomes computationally tractable). Our toy task uses a simple 1D environment, where the agent starts in a random position between $[-2, 2]$ and has to reach a goal location that is randomly sampled at the beginning of each episode in $[-1, 1]$. We parameterize the Gaussian policy as an MLP with two hidden layers that contains 400 parameters. Once the policy is partially trained, we estimate the ground-truth gradient using a large number of rollouts and compare them with the gradients estimated using each technique. We perform this comparison across a range of sample sizes used to obtain the latter estimate. Figure 4 shows the Pearson correlation between the estimated and 'ground-truth' gradients at each sample size for each method. Gradients estimated using TMAML are consistently best correlated with the ground-truth (even when using only a small number of samples, where other methods produce only weak correlations), further illustrating the high quality gradient estimation produced by our technique.

5.4. Baseline Comparisons

To further demonstrate the benefit afforded by TMAML in terms of task performance, we compare our methods to existing ones across the set of baseline tasks described above. The mean reward as a function of training iterations is plotted for each task and method in Figure 3. From these results, we can see that TMAML with meta-learned control variates and TMAML with per-task control variates improve upon MAML by a substantial margin and also typically outperform both LVC and DICE. This result is most consis-

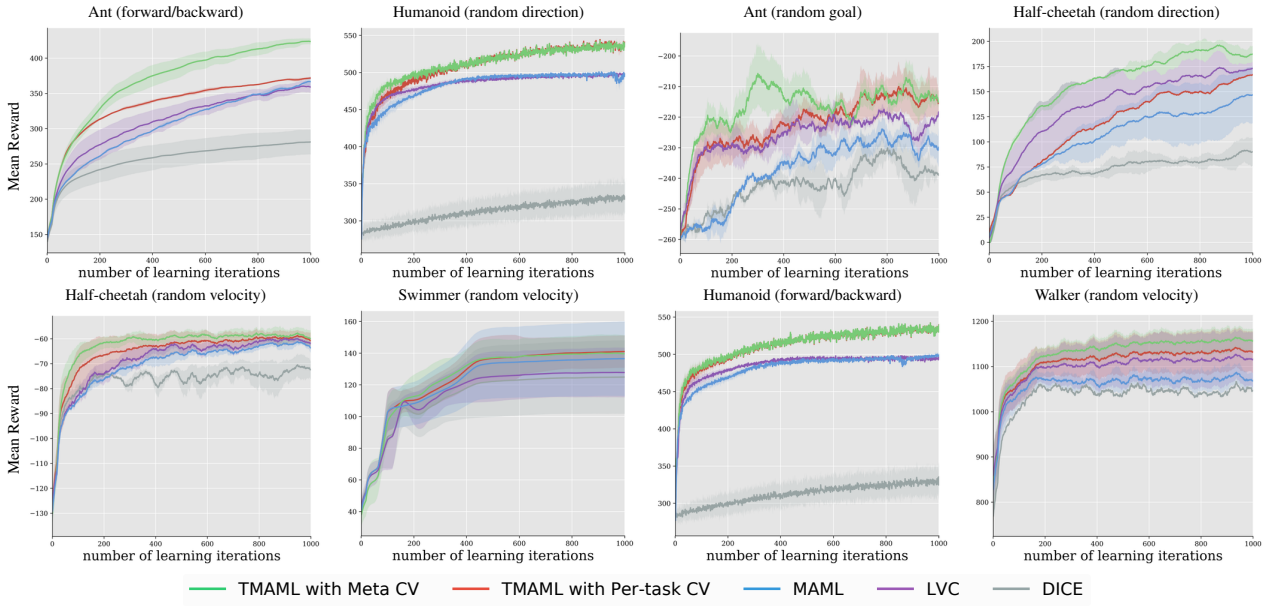

Figure 3. Comparison of TMAML against MAML, LVC, and DICE across a set of Meta-RL environments. Shaded area represents one standard deviation, learning curves are on 5 random seeds.

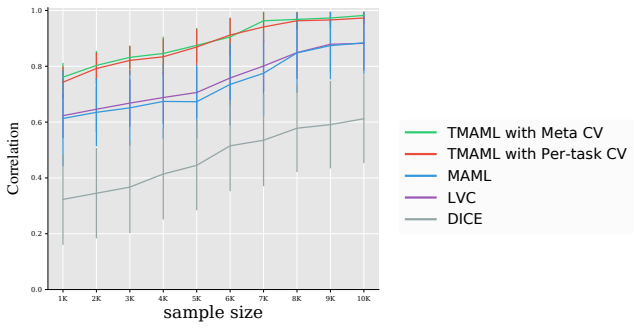

Figure 4. Correlation between 'ground-truth' gradients and gradients estimated using each technique in a 1D environment.

tent for the version of TMAML with meta-learned control variates, where performance either matches or exceeds the per-task counterpart across each benchmark. For our more exhaustive comparison shown in Figure 3, we train MAML, DICE, and LVC with learning algorithm VPG for the sake of fair comparison. From our observations, PPO generally performs better than VPG, to study the effect of learning algorithm choice, we include results on a pair of the baseline tasks for each of the methods when using PPO. Final task performance on these experiments is shown in Table 1. We observe a general trend towards better performance when using PPO and point out that TMAML continues to outperform other methods. Taken together, these results clearly demonstrate that TMAML achieves better gradient estimation, which directly translates into improved performance in challenging Meta-RL environments.

	Ant (forward/backward)		Humanoid (random direction)	
Method	Meta CV	Per-Task CV	Meta CV	Per-Task CV
TMAML	**513 ± 21.3**	487 ± 23.9	**557 ± 19.6**	547 ± 16.7
LVC	386 ± 14.6		482 ± 18.3	
MAML	401 ± 24.4		452 ± 27.9	
DICE	281 ± 46.7		397 ± 52.3	

Table 1. Results of different control variates and methods for estimating Meta-RL gradient, when combined with PPO. The reported results are the average reward at the 1000-th learning iteration.

6. Conclusion

We propose a surrogate objective function named TMAML that adds control variates into gradient estimation via automatic differentiation frameworks. We show analytically and empirically that TMAML reduces gradient variance without introducing bias and illustrate its improved sample complexity and better asymptotic performance across a set of challenging Meta-RL environments. In addition, we introduce a method to meta-learn the control variates themselves, enabling efficient and scalable learning from a distribution of MDPs. TMAML is easy to implement and computationally efficient with automatic differentiation, it can be conveniently combined with other gradient-based Meta-RL algorithms to improve sample efficiency. In multi-agent reinforcement learning, higher order gradients are common and are often high variance, which impedes learning and makes the application of TMAML with meta control variates an attractive avenue for future research.

Acknowledgments

The authors would like to thank Alexander Trott and Lily Hu for their valuable discussions and the anonymous reviewers for their helpful comments. The authors also thank the team members at Salesforce Research for their support.

References

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations*, 2018.

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pp. 3981–3989, 2016.

Andrychowicz, M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Chen, Y., Hoffman, M. W., Colmenarejo, S. G., Denil, M., Lillicrap, T. P., Botvinick, M., and de Freitas, N. Learning to learn without gradient descent by gradient descent. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 748–756. JMLR. org, 2017.

Clavera, I., Rothfuss, J., Schulman, J., Fujita, Y., Asfour, T., and Abbeel, P. Model-based reinforcement learning via meta-policy optimization. In Billard, A., Dragan, A., Peters, J., and Morimoto, J. (eds.), *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pp. 617–629. PMLR, 29–31 Oct 2018.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Foerster, J., Farquhar, G., Al-Shedivat, M., Rocktäschel, T., Xing, E., and Whiteson, S. DiCE: The infinitely differentiable Monte Carlo estimator. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1529–1538, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

Greensmith, E., Bartlett, P. L., and Baxter, J. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov): 1471–1530, 2004.

Gu, S., Holly, E., Lillicrap, T., and Levine, S. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3389–3396. IEEE, 2017.

Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, pp. 2944–2952, 2015.

Heess, N., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, A., Riedmiller, M., et al. Emergence of locomotion behaviours in rich environments. *Advances in Neural Information Processing Systems(NeurIPS)*, 2017.

Liang, C., Norouzi, M., Berant, J., Le, Q., and Lao, N. Memory augmented policy optimization for program synthesis with generalization. *Advances in Neural Information Processing Systems(NeurIPS)*, 2018.

Liu, H., Feng, Y., Mao, Y., Zhou, D., Peng, J., and Liu, Q. Action-depedent control variates for policy optimization via stein's identity. *arXiv preprint arXiv:1710.11198*, 2017.

Liu, H., Trott, A., Socher, R., and Xiong, C. Competitive experience replay. In *International Conference on Learning Representations*, 2019.

Mao, H., Venkatakrishnan, S. B., Schwarzkopf, M., and Alizadeh, M. Variance reduction for reinforcement learning in input-driven environments. In *International Conference on Learning Representations*, 2019a.

Mao, J., Foerster, J., Rocktäschel, T., Al-Shedivat, M., Farquhar, G., and Whiteson, S. A baseline for any order gradient estimation in stochastic computation graphs. In

Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4343–4351, Long Beach, California, USA, 09–15 Jun 2019b. PMLR.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *International Conference of Learning Representations (ICLR)*, 2018.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529, 2015.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *CoRR, abs/1803.02999*, 2, 2018.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. *International Conference of Learning Representations (ICLR)*, 2016.

Rothfuss, J., Lee, D., Clavera, I., Asfour, T., and Abbeel, P. ProMP: Proximal meta-policy search. In *International Conference on Learning Representations*, 2019.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *International Conference on Machine Learning*, pp. 1312–1320, 2015.

Schmidhuber, J. *Evolutionary principles in self-referential learning*. PhD thesis, Technische Universität München, 1987.

Schulman, J., Heess, N., Weber, T., and Abbeel, P. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, pp. 3528–3536, 2015.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *Advances in Neural Information Processing Systems*, 2017.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

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

Stadie, B., Yang, G., Houthooft, R., Chen, P., Duan, Y., Wu, Y., Abbeel, P., and Sutskever, I. The importance of sampling inmeta-reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 9280–9290, 2018.

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

Thomas, P. Bias in natural actor-critic algorithms. In *International Conference on Machine Learning*, pp. 441–448, 2014.

Thrun, S. and Pratt, L. *Learning to learn*. Springer Science & Business Media, 2012.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 5026–5033. IEEE, 2012.

Tucker, G., Bhupatiraju, S., Gu, S., Turner, R., Ghahramani, Z., and Levine, S. The mirage of action-dependent baselines in reinforcement learning. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5015–5024, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Vinyals, O., Babuschkin, I., Chung, J., Mathieu, M., Jaderberg, M., Czarnecki, W. M., Dudzik, A., Huang, A., Georgiev, P., Powell, R., Ewalds, T., Horgan, D., Kroiss, M., Danihelka, I., Agapiou, J., Oh, J., Dalibard, V., Choi, D., Sifre, L., Sulsky, Y., Vezhnevets, S., Molloy, J., Cai, T., Budden, D., Paine, T., Gulcehre, C., Wang, Z., Pfaff, T., Pohlen, T., Wu, Y., Yogatama, D., Cohen, J., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Apps, C., Kavukcuoglu, K., Hassabis, D., and Silver, D. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starc 2019.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. Learning to reinforcement learn, 2016.

Weber, T., Heess, N., Buesing, L., and Silver, D. Credit assignment techniques in stochastic computation graphs. *arXiv preprint arXiv:1901.01761*, 2019.

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

Wu, C., Rajeswaran, A., Duan, Y., Kumar, V., Bayen, A. M., Kakade, S., Mordatch, I., and Abbeel, P. Variance reduction for policy gradient with action-dependent factorized baselines. In *International Conference on Learning Representations*, 2018.

Xu, T., Liu, Q., Zhao, L., and Peng, J. Learning to explore via meta-policy gradient. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5463–5472, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018a. PMLR.

Xu, Z., van Hasselt, H. P., and Silver, D. Meta-gradient reinforcement learning. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 2396–2407. Curran Associates, Inc., 2018b.