ProMP: Proximal Meta-Policy Search

Jonas Rothfuss*
UC Berkeley, KIT
jonas.rothfuss@gmail.com

Dennis Lee*, **Ignasi Clavera***
UC Berkeley
{dennisl88,iclavera}@berkeley.edu

Tamim Asfour
Karlsruhe Inst. of Technology (KIT)

Pieter Abbeel
UC Berkeley, Covariant.ai

Abstract

Credit assignment in meta-reinforcement learning (Meta-RL) is still poorly understood. Existing methods either neglect credit assignment to pre-adaptation behavior or implement it naively. This leads to poor sample-efficiency during meta-training as well as ineffective task identification strategies. In this work, we provide a theoretical analysis of credit assignment in gradient-based Meta-RL. Building on the gained insights we develop a novel meta-learning algorithm that overcomes both the issue of poor credit assignment and previous difficulties in estimating meta-policy gradients. Our approach leads to superior pre-adaptation policy behavior and consistently outperforms previous Meta-RL algorithms.

1 Sampling Distribution Credit Assignment in Meta-RL

Meta-Reinforcement Learning aims to learn a learning algorithm which is able to quickly learn the optimal policy for a task (MDP) $\mathcal{T}$ drawn from a distribution of tasks $\rho(\mathcal{T})$. Meta-RL is a multi-stage process in which the agent, after a few sampled environment interactions, adapts its behavior to the given task. Despite its wide utilization, little work has been done to promote theoretical understanding of this process, leaving Meta-RL grounded on unstable foundations. Although the behavior prior to the adaptation step is instrumental for task identification, the interplay between pre-adaptation sampling and posterior performance of the policy remains poorly understood. In fact, prior work in gradient-based Meta-RL has either entirely neglected credit assignment to the pre-update distribution [4] or implemented such credit assignment in a naive way [1, 12].

Gradient-based meta-learning approaches perform Meta-RL by learning the parameters θ of a policy π_θ such that performing a single or few steps of vanilla policy gradient (VPG) with the given task leads to the optimal policy for that task. This meta-learning formulation, also known under the name of MAML, was first introduced by [4]. We refer to it as formulation I and can be expressed as maximizing the objective

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

In that, U denotes the update function which depends on the task $\mathcal{T}$, and performs one VPG step towards maximizing the performance of the policy in $\mathcal{T}$. Later work proposes a slightly different notion of gradient-based Meta-RL, also known as E-MAML, that attempts to circumvent issues with the meta-gradient estimation in MAML [1, 12]:

$$J^{II}(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\left[\mathbb{E}_{\substack{\boldsymbol{\tau}^{1:N}\sim P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta)\\ \boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}}\left[R(\boldsymbol{\tau}')\right]\right] \text{ with } \theta' := U(\theta,\boldsymbol{\tau}^{1:N}) = \theta + \alpha\nabla_\theta \sum_{n=1}^{N}\left[R(\boldsymbol{\tau}^{(n)})\right]$$

Formulation II views U as a deterministic function that depends on N sampled trajectories from a specific task. In contrast to formulation I, the expectation over pre-update trajectories $\boldsymbol{\tau}$ is applied outside of the update function. Formulation I (Fig. 1 left) propagates the credit assignment through

*authors contributed equally to this work

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

the update step, thereby exploiting the full problem structure. In contrast, formulation II (Fig. 1 right) neglects the inherent structure, directly assigning credit from post-update return R' to the pre-update policy π_θ which leads to noisier, less effective credit assignment.

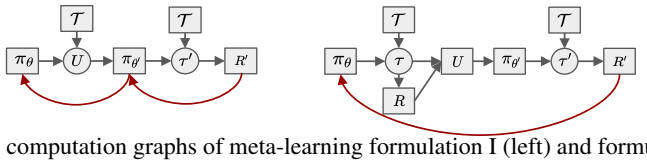

Figure 1: Stochastic computation graphs of meta-learning formulation I (left) and formulation II (right). The red arrows illustrate the credit pre-update sampling distribution assignment

Both formulations optimize for the same objective, and are equivalent at the 0^{th} order. However, because of the difference in their formulation, their gradients and the resulting optimization step differs. In the following, we shed light on how and where formulation II loses signal by analyzing the gradients of both formulations, which can be written as (see Appendix A for derivations)

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\left[\mathbb{E}_{\substack{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)\\ \boldsymbol{\tau}'\sim P_\mathcal{T}(\boldsymbol{\tau}'|\theta')}}\left[\nabla_\theta J_{\text{post}}(\boldsymbol{\tau},\boldsymbol{\tau}') + \nabla_\theta J_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}')\right]\right] \tag{1}$$

The first term $\nabla_\theta J_{\text{post}}(\boldsymbol{\tau},\boldsymbol{\tau}')$ is equal in both formulations, but the second term, $\nabla_\theta J_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}')$, differ:

$$\nabla_\theta J^{II}_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}') = \alpha\nabla_\theta \log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau}') \tag{2}$$

$$\nabla_\theta J^{I}_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}') = \alpha\nabla_\theta \log\pi_\theta(\boldsymbol{\tau})\Bigg(\underbrace{(\nabla_\theta \log\pi_\theta(\boldsymbol{\tau})R(\boldsymbol{\tau}))^\top}_{\nabla_\theta J^{\text{inner}}}\underbrace{(\nabla_{\theta'} \log\pi_{\theta'}(\boldsymbol{\tau}')R(\boldsymbol{\tau}'))}_{\nabla_{\theta'} J^{\text{outer}}}\Bigg) \tag{3}$$

The credit assignment w.r.t. the pre-updated sampling distribution is carried out by the second term. In formulation II, $\nabla_\theta J^{II}_{\text{pre}}$ can be viewed as standard reinforcement learning on π_θ with $R(\boldsymbol{\tau}')$ as reward signal, treating the update function U as part of the unknown dynamics of the system. This shifts the pre-update sampling distribution to better adaptation steps. Formulation I takes the causal dependence of $P_\mathcal{T}(\boldsymbol{\tau}'|\theta')$ on $P_\mathcal{T}(\boldsymbol{\tau}|\theta)$ into account. It does so by maximizing the inner product of pre-update and post-update policy gradients (see Eq. 3). This steers the pre-update policy towards 1) larger post-updates returns 2) larger adaptation steps $\alpha\nabla_\theta J^{\text{inner}}$, 3) better alignment of pre- and post-update policy gradients. When combined, these effects directly optimize for adaptation. As a result, we expect the first meta-policy gradient formulation, J^I, to yield superior learning properties.

2 Low Variance Curvature Estimator

In the previous section we show that the formulation J^I introduced by [4] results in superior meta-gradient updates, which should in principle lead to improved convergence properties. As we show in Appendix A.1, we can write the gradient of the meta-learning objective as

$$\nabla_\theta J^I(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\Big[\mathbb{E}_{\boldsymbol{\tau}'\sim P_\mathcal{T}(\boldsymbol{\tau}'|\theta')}\left[\nabla_{\theta'}\log P_\mathcal{T}(\boldsymbol{\tau}'|\theta')R(\boldsymbol{\tau}')\nabla_\theta U(\theta,\mathcal{T})\right]\Big] \tag{4}$$

Since the update function U resembles a policy gradient step, its gradient $\nabla_\theta U(\theta,\mathcal{T})$ involves computing the hessian of the reinforcement learning objective, i.e., $\nabla^2_\theta\,\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right]$. The expectation of the RL-objective is in general intractable its gradients are typically computed using a score function Monte Carlo estimator [15, 13]. However, score function surrogate objectives yield wrong higher order derivatives, resulting in strongly biased estimates of the RL-objective hessian. This can be overcome using DiCE formulation [5], but this leads to high variance estimates. To facilitate a sample efficient meta-learning, we introduce the low variance curvature (LVC) estimator:

$$J^{\text{LVC}}(\boldsymbol{\tau}) = \sum_{t=0}^{H-1}\frac{\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\perp(\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t))}\left(\sum_{t'=t}^{H-1} r(\boldsymbol{s}_{t'},\boldsymbol{a}_{t'})\right)\quad \boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}) \tag{5}$$

When compared to the DiCE, this estimator neglects the sequential dependence of $\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)$ within trajectories, which leads to a variance reduction, but makes the estimate biased. The choice of this objective function is motivated by findings in [6]: under certain conditions the bias of J^{LVC} vanishes around local optima. The experiments in section 4.2 underpin the theoretical findings, showing that the low variance hessian estimates obtained through J^{LVC} improve the sample-efficiency of meta-learning by a significant margin when compared to J^{DiCE}. We refer the interested reader to Appendix B for derivations and a more detailed discussion.

3 ProMP: Proximal Meta-Policy Search

Building on the previous sections, we develop a novel meta-policy search method based on the low variance curvature objective which aims to solve the Meta-RL formulation J^I. To do so, we build upon the recently introduced PPO algorithm [11], which achieves comparable results to TRPO with the advantage of being a first order method. PPO uses a surrogate clipping objective J^{CLIP} which allows it to safely take multiple gradient steps without re-sampling trajectories.

In case of Meta-RL, it does not suffice to just replace the post-update reward objective with $J_{\mathcal{T}}^{\text{CLIP}}$. In order to safely perform multiple meta-gradient steps based on the same sampled data from a recent policy π_{θ_o}, we also need to 1) account for changes in the pre-update action distribution $\pi_\theta(a_t|s_t)$, and 2) bound changes in the pre-update state visitation distribution [7].

We propose Proximal Meta-Policy Search (ProMP) which incorporates both the benefits of proximal policy optimization and the low variance curvature objective (see Alg. 1.) In order to comply with requirement 1), ProMP replaces the "stop gradient" importance weight $\frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))}$ by the likelihood ratio $\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_o}(a_t|s_t))}$, which results in the following objective

$$J_{\mathcal{T}}^{LR}(\theta) = \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau},\theta_o)}\left[\sum_{t=0}^{H-1} \frac{\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}{\pi_{\theta_o}(\boldsymbol{a}_t|\boldsymbol{s}_t)} A^{\pi_{\theta_o}}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{6}$$

An important feature of this objective is that its derivatives w.r.t θ evaluated at θ_o are identical to those of the LVC objective, and it additionally accounts for changes in the pre-update action distribution. To satisfy condition 2) we extend the clipped meta-objective with a KL-penalty term between π_θ and π_{θ_o}. This KL-penalty term enforces a soft local "trust region" around π_{θ_o}, preventing the shift in state visitation distribution to become large during optimization. This enables us to take multiple meta-policy gradient steps without re-sampling. Altogether, ProMP optimizes

$$J_{\mathcal{T}}^{\text{ProMP}}(\theta) = J_{\mathcal{T}}^{\text{CLIP}}(\theta') - \eta\bar{\mathcal{D}}_{KL}(\pi_{\theta_o}, \pi_\theta) \quad \text{s.t.} \quad \theta' = \theta + \alpha\, \nabla_\theta J_{\mathcal{T}}^{LR}(\theta)\,, \quad \mathcal{T}\sim\rho(\mathcal{T}) \tag{7}$$

ProMP consolidates the insights developed throughout the course of this paper, while at the same time making maximal use of recently developed policy gradients algorithms. First, its meta-learning formulation exploits the full structural knowledge of gradient-based meta-learning. Second, it incorporates a low variance estimate of the RL-objective hessian. Third, ProMP controls the statistical distance of both pre- and post-adaptation policies, promoting efficient and stable meta-learning. All in all, ProMP consistently outperforms previous gradient-based meta-RL algorithms in sample complexity, wall clock time, and asymptotic performance (see Section 4.1).

4 Experiments

In order to empirically validate the theoretical arguments outlined above, this section provides a experimental analysis that aims to answer the following questions: (i) How does ProMP perform against previous Meta-RL algorithms? (ii) How do the lower variance but biased LVC gradient estimates compare to the high variance, unbiased DiCE estimates? (iii) Do the different formulations result in different pre-update exploration properties? The source code and the experiments data are available on our supplementary website.[2]

4.1 Meta-Gradient Based Comparison

We compare our method, ProMP, in sample complexity and asymptotic performance to four other gradient-based approaches: TRPO-MAML [4], E-MAML-TRPO, E-MAML-VPG [12], and LVC-VPG, an ablated version of our method that uses the LVC objective in the adaptation step and meta-optimizes with vanilla policy gradient. These algorithms are benchmarked on three different locomotion tasks [3, 14] that require adaptation: the half-cheetah must switch between running forward and backward, the quadruped agent ant must run in different directions in the 2D-plane, and the hopper has adapt to different configuration of their dynamics.

The results (Fig. 2) highlight the strength of ProMP in terms of sample efficiency and final performance. They also demonstrate the positive effect of the LVC objective: LVC-VPG, even though optimized with vanilla policy gradient, is often able to achieve comparable results to the the prior methods that are optimized with TRPO. When compared to E-MAML-VPG, LVC proves superior in performance which underpins the soundness of the theory developed throughout this paper.

[2] https://sites.google.com/view/pro-mp

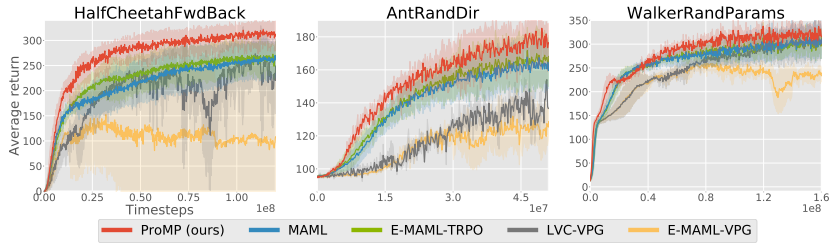

Figure 2: Meta-learning curves of ProMP and four other gradient-based meta-learning algorithms in six different Mujoco environments. ProMP outperforms previous work in all the the environments.

4.2 Estimator Variance and Its Effect on Meta-Learning

In Section 2 we discussed how the DiCE formulation yields unbiased but high variance estimates of the RL-objective hessian and served as motivation for the low variance curvature (LVC) estimator. Here we investigate the meta-gradient variance of both estimators as well as its implication on the learning performance. Specifically, we report the relative standard deviation of the meta-policy gradients as well as the average return throughout the learning process in the HalfCheetahFwdBack environment. The results, depicted in Figure 3, highlight the advantage of the low variance curvature estimate. The trajectory level dependencies inherent in the DiCE estimator lead to a meta-gradient standard deviation that is on average two times higher when compared to LVC. As the learning curves indicate, the noisy gradients impede sample efficient meta-learning in case of DiCE. Meta-policy search based on the LVC estimator leads to substantially better learning properties.

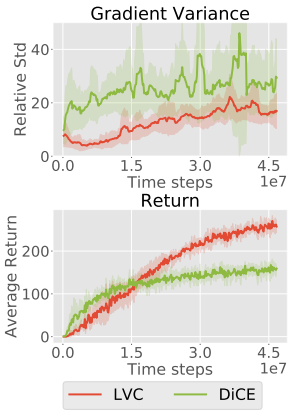

Figure 3: Upper: Relative standard deviation of meta-policy gradients. Lower: Return in the HalfCheetahFwdBack env.

4.3 Comparison of Initial Sampling Distributions

Here we evaluate the effect of the different objectives on the learned pre-update sampling distribution. We compare the low variance curvature (LVC) estimator with TRPO (LVC-TRPO) against MAML [4] and E-MAML-TRPO [12] in a 2D environment on which the exploration behavior can be visualized. Each task of this environment corresponds to reaching a different corner location; however, the 2D agent only experiences reward when it is sufficiently close to the corner (translucent regions of Figure 4). Thus, to successfully identify the task, the agent must explore the different regions. We perform three inner adaptation steps on each task, allowing the agent to fully change its behavior from exploration to exploitation.

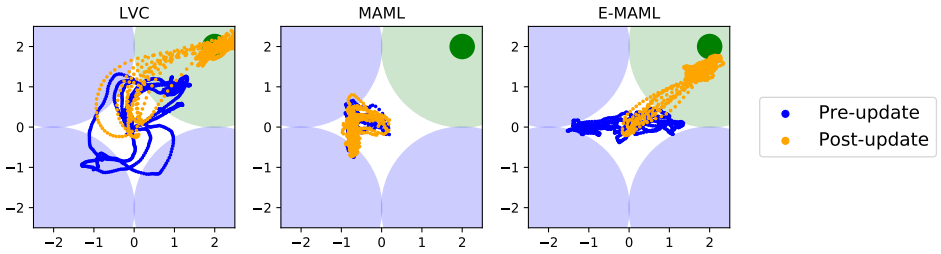

Figure 4: Exploration patterns of the pre-update policy and exploitation post-update with different update functions. Through its superior credit assignment, the LVC objective learns a pre-update policy that is able to identify the current task and respectively adapt its policy, successfully reaching the goal (dark green circle).

The different exploration-exploitation strategies are displayed in Figure 4. Since the MAML implementation does not assign credit to the pre-update sampling trajectory, it is unable to learn a sound exploration strategy for task identification and thus fails to accomplish the task. On the other hand, E-MAML, which corresponds to formulation II, learns to explore in long but random paths: because it can only assign credit to batches of pre-update trajectories, there is no notion of which actions in particular facilitate good task adaptation. As consequence the adapted policy slightly misses the task-specific target. The LVC estimator, instead, learns a consistent pattern of exploration, visiting each of the four regions, which it harnesses to fully solve the task.

Acknowledgments

Ignasi Clavera was supported by the La Caixa Fellowship. The research leading to these results has received funding from the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft) under Priority Program on Autonomous Learning (SPP 1527) and was supported by Berkeley Deep Drive, Amazon Web Services, and Huawei.

References

[1] Maruan Al-Shedivat, Trapit Bansal, Umass Amherst, Yura Burda, Openai Ilya, Sutskever Openai, Igor Mordatch Openai, and Pieter Abbeel. Continuous Adaptation via Meta-Learning in Nonstationary and Competitive Environments. In *ICLR*, 2018.

[2] Jonathan Baxter and Peter L Bartlett. Infinite-Horizon Policy-Gradient Estimation. Technical report, 2001.

[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. Technical report, 6 2016.

[4] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *ICML*, 2017.

[5] Jakob Foerster, Gregory Farquhar, Maruan Al-Shedivat, Tim Rocktäschel, Eric P Xing, and Shimon Whiteson. DiCE: The Infinitely Differentiable Monte Carlo Estimator. In *ICML*, 2018.

[6] Thomas Furmston, Guy Lever, David Barber, and Joelle Pineau. Approximate Newton Methods for Policy Search in Markov Decision Processes. Technical report, 2016.

[7] Sham Kakade and John Langford. Approximately Optimal Approximate Reinforcement Learning. In *ICML*, 2002.

[8] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to Generalize: Meta-Learning for Domain Generalization. In *AAAI*, 2017.

[9] Alex Nichol, Joshua Achiam, and John Schulman. On First-Order Meta-Learning Algorithms. Technical report, 2018.

[10] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient Estimation Using Stochastic Computation Graphs. In *NIPS*, 2015.

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov Openai. Proximal Policy Optimization Algorithms. *CoRR*, 2017.

[12] Bradly C Stadie, Ge Yang, Rein Houthooft, Xi Chen, Yan Duan, Yuhuai Wu, Pieter Abbeel, and Ilya Sutskever. Some Considerations on Learning to Explore via Meta-Reinforcement Learning. Technical report, 2018.

[13] Richard S. Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *NIPS*, 2000.

[14] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IROS*, pages 5026–5033. IEEE, 10 2012.

[15] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 5 1992.

A Two Meta-Policy Gradient Formulations

In this section we discuss two different gradient-based meta-learning formulations, derive their gradients and analyze the differences between them.

A.1 Meta-Policy Gradient Formulation I

The first meta-learning formulation, known as MAML [4], views the inner update rule $U(\theta, \mathcal{T})$ as a mapping from the pre-update parameter θ and the task $\mathcal{T}$ to an adapted policy parameter θ'. The update function can be viewed as stand-alone procedure that encapsulates sampling from the task-specific trajectory distribution $P_{\mathcal{T}}(\boldsymbol{\tau}|\pi_\theta)$ and updating the policy parameters. Building on this concept, the meta-objective can be written as

$$J^I(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\left[\mathbb{E}_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}\left[R(\boldsymbol{\tau}')\right]\right] \quad \text{with} \quad \theta' := U(\theta, \mathcal{T}) \tag{8}$$

The task-specific gradients follow as

$$\begin{aligned}
\nabla_\theta J^I_{\mathcal{T}}(\theta) &= \nabla_\theta E_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}\left[R(\boldsymbol{\tau}')\right] && (9)\\
&= E_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}\left[\nabla_\theta \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') R(\boldsymbol{\tau}')\right] && (10)\\
&= E_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}\left[\nabla_{\theta'} \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') R(\boldsymbol{\tau}')\nabla_\theta \theta'\right] && (11)
\end{aligned}$$

In order to derive the gradients of the inner update $\nabla_\theta \theta' = \nabla_\theta U(\theta, \mathcal{T})$ it is necessary to know the structure of U. The main part of this paper assumes the inner update rule to be a policy gradient descent step

$$\begin{aligned}
\nabla_\theta U(\theta, \mathcal{T}) &= \nabla_\theta \left(\theta + \alpha\, \nabla_\theta \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right]\right) && (12)\\
&= I + \alpha \nabla^2_\theta\, \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right] && (13)
\end{aligned}$$

Thereby the second term in (13) is the local curvature (hessian) of the inner adaptation objective function. The correct hessian of the inner objective can be derived as follows:

$$\begin{aligned}
\nabla^2_\theta\, \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right] &= \nabla_\theta\, \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) R(\boldsymbol{\tau})\right] && (14)\\
&= \nabla_\theta \int P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau} && (15)\\
&= \int P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau})^\top R(\boldsymbol{\tau}) + && (16)\\
&\qquad P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \nabla^2_\theta \log \pi_\theta(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau} && (17)\\
&= \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\left(\nabla^2_\theta \log \pi_\theta(\boldsymbol{\tau}) + \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau})^\top\right)\right] && (18)
\end{aligned}$$

A.2 Meta-Policy Gradient Formulation II

The second meta-reinforcement learning formulation views the the inner update $\theta' = U(\theta, \tau^{1:N})$ as a deterministic function of the pre-update policy parameters θ and N trajectories $\boldsymbol{\tau}^{1:N} \sim P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta)$ sampled from the pre-update trajectory distribution. This formulation was introduced in [1] and further discussed with respect to its exploration properties in [12].

Viewing U as a function that adapts the policy parameters θ to a specific task $\mathcal{T}$ given policy rollouts in this task, the corresponding meta-learning objective can be written as

$$J^{II}(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\left[\mathbb{E}_{\boldsymbol{\tau}^{1:N}\sim P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta)}\left[\mathbb{E}_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}\left[R(\boldsymbol{\tau}')\right]\right]\right] \quad \text{with } \theta' := U(\theta, \boldsymbol{\tau}^{1:N}) \tag{19}$$

Since the first part of the gradient derivation is agnostic to the inner update rule $U(\theta, \tau^{1:N})$, we only assume that the inner update function U is differentiable w.r.t. θ. First we rewrite the meta-objective $J(\theta)$ as expectation of task specific objectives $J^{II}_{\mathcal{T}}(\theta)$ under the task distribution. This allows us to express the meta-policy gradients as expectation of task-specific gradients:

$$\nabla_\theta J^{II}(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})}\left[\nabla_\theta J^{II}_{\mathcal{T}}(\theta)\right] \tag{20}$$

The task specific gradients can be calculated as follows

$$\begin{aligned}
\nabla_\theta J_{\mathcal{T}}^{II}(\theta) = \quad & \nabla_\theta \mathbb{E}_{\boldsymbol{\tau} \sim P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta)} \Big[\mathbb{E}_{\boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')} \big[R(\boldsymbol{\tau}') \big] \Big] \\
= \quad & \nabla_\theta \int \int R(\boldsymbol{\tau}') \; P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') \; P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta) \; d\boldsymbol{\tau}' \; d\boldsymbol{\tau} \\
= \quad & \mathbb{E}_{\substack{\boldsymbol{\tau}^{1:N} \sim P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta) \\ \boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}} \left[R(\boldsymbol{\tau}') \left(\nabla_\theta \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') + \sum_{i=1}^{N} \nabla_\theta \log P_{\mathcal{T}}(\boldsymbol{\tau}^{(n)}|\theta) \right) \right] \\
= \quad & \mathbb{E}_{\substack{\boldsymbol{\tau}^{1:N} \sim P_{\mathcal{T}}(\boldsymbol{\tau}^{1:N}|\theta) \\ \boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}} \left[R(\boldsymbol{\tau}') \left(\nabla_{\theta'} \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') \nabla_\theta \theta' + \sum_{n=1}^{N} \nabla_\theta \log P_{\mathcal{T}}(\boldsymbol{\tau}^{(n)}|\theta) \right) \right]
\end{aligned}$$

As in A.1 the structure of $U(\theta, \boldsymbol{\tau}^{1:N})$ must be known in order to derive the gradient $\nabla_\theta \theta'$. Since we assume the inner update to be vanilla policy gradient, the respective gradient follows as

$$U(\theta, \tau^{1:N}) = \theta + \alpha \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \log \pi_\theta(\tau^{(n)})) R(\tau^{(n)}) \quad \text{with} \quad \nabla_\theta \log \pi_\theta(\tau) = \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

The respective gradient of $U(\theta, \tau^{1:N})$ follows as

$$\nabla_\theta U(\theta, \tau^{1:N}) = \nabla_\theta \left(\theta + \alpha \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \log \pi_\theta(\tau^{(n)})) R(\tau^{(n)}) \right) \tag{21}$$

$$= I + \alpha \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta^2 \log \pi_\theta(\tau^{(n)})) R(\tau^{(n)}) \tag{22}$$

A.3 Comparing the Gradients of the Two Formulations

In the following we analyze the differences between the gradients derived for the two formulations. To do so, we begin with $\nabla_\theta J_{\mathcal{T}}^{I}(\theta)$ by inserting the gradient of the inner adaptation step (13) into (11):

$$\nabla_\theta J_{\mathcal{T}}^{I}(\theta) = E_{\boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')} \left[\nabla_{\theta'} \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') R(\boldsymbol{\tau}') \left(I + \alpha \nabla_\theta^2 \, \mathbb{E}_{\boldsymbol{\tau} \sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)} \left[R(\boldsymbol{\tau}) \right] \right) \right] \tag{23}$$

We can substitute the hessian of the inner objective by its derived expression from (18) and then rearrange the terms. Also note that $\nabla_\theta \log P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) = \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) = \sum_{t=1}^{H-1} \log \pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)$ where H is the MDP horizon.

$$\nabla_\theta J_{\mathcal{T}}^{I}(\theta) = E_{\boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')} \Bigg[\nabla_{\theta'} \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') R(\boldsymbol{\tau}') \Bigg(I + \alpha \mathbb{E}_{\boldsymbol{\tau} \sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)} \big[R(\boldsymbol{\tau}) \tag{24}$$

$$\big(\nabla_\theta^2 \log \pi_\theta(\boldsymbol{\tau}) + \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau})^\top \big) \big] \Bigg) \Bigg] \tag{25}$$

$$= \mathbb{E}_{\substack{\boldsymbol{\tau} \sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \\ \boldsymbol{\tau}' \sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}} \Bigg[\underbrace{\nabla_{\theta'} \log \pi_{\theta'}(\boldsymbol{\tau}') R(\boldsymbol{\tau}') \Big(I + \alpha R(\boldsymbol{\tau}) \nabla_\theta^2 \log \pi_\theta(\boldsymbol{\tau}) \Big)}_{\nabla_\theta J_{\text{post}}(\boldsymbol{\tau}, \boldsymbol{\tau}')} \tag{26}$$

$$\underbrace{+ \alpha \nabla_{\theta'} \log \pi_{\theta'}(\boldsymbol{\tau}') R(\boldsymbol{\tau}') R(\boldsymbol{\tau}) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau})^\top}_{\nabla_\theta J_{\text{pre}}^{I}(\boldsymbol{\tau}, \boldsymbol{\tau}')} \Bigg] \tag{27}$$

Next, we rearrange the gradient of J^{II} into a similar form as $\nabla_\theta J_{\mathcal{T}}^{I}(\theta)$. For that, we start by inserting (22) for $\nabla_\theta \theta'$ and replacing the expectation over pre-update trajectories $\boldsymbol{\tau}^{1:N}$ by the expectation over

a single trajectory $\boldsymbol{\tau}$.

$$\nabla_\theta J^I_{\mathcal{T}}(\theta) = \mathbb{E}_{\substack{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \\ \boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')}} \Bigg[\underbrace{R(\boldsymbol{\tau}')\nabla_{\theta'} \log \pi_\theta(\boldsymbol{\tau}') \left(I + \alpha R(\boldsymbol{\tau}) \nabla^2_\theta \log \pi_\theta(\tau)\right)}_{\nabla_\theta J_{\text{post}}(\boldsymbol{\tau},\boldsymbol{\tau}')} \tag{28}$$

$$\underbrace{+R(\boldsymbol{\tau}')\nabla_\theta \log \pi_\theta(\boldsymbol{\tau})}_{\nabla_\theta J^I_{\text{pre}}(\boldsymbol{\tau},\boldsymbol{\tau}')} \Bigg] \tag{29}$$

While the first part of the gradients match ((26) and (28)), the second part ((27) and (29)) differs. Since the second gradient term can be viewed as responsible for shifting the pre-update sampling distribution $P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)$ towards higher post-update returns, we refer to it as $\nabla_\theta J_{\text{pre}}(\boldsymbol{\tau}, \boldsymbol{\tau}')$. To further analyze the difference between $\nabla_\theta J^I_{\text{pre}}$ and $\nabla_\theta J^{II}_{\text{pre}}$ we slightly rearrange (27) and put both gradient terms next to each other:

$$\nabla_\theta J^I_{\text{pre}}(\boldsymbol{\tau}, \boldsymbol{\tau}') = \alpha \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) \left(\underbrace{\left(\nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) R(\boldsymbol{\tau})\right)^\top}_{\nabla_\theta J^{\text{inner}}} \underbrace{\left(\nabla_{\theta'} \log \pi_{\theta'}(\boldsymbol{\tau}') R(\boldsymbol{\tau}')\right)}_{\nabla_{\theta'} J^{\text{outer}}} \right) \tag{30}$$

$$\nabla_\theta J^{II}_{\text{pre}}(\boldsymbol{\tau}, \boldsymbol{\tau}') = \alpha \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) R(\boldsymbol{\tau}') \tag{31}$$

In the following we interpret and and compare of the derived gradient terms, aiming to provide intuition for the differences between the formulations:

The first gradient term J_{post} that matches in both formulations corresponds to a policy gradient step on the post-update policy $\pi_{\theta'}$. Since θ' itself is a function of θ, the term $\left(I + \alpha R(\boldsymbol{\tau}) \nabla^2_\theta \log \pi_\theta(\tau)\right)$ can be seen as linear transformation of the policy gradient update $R(\boldsymbol{\tau}')\nabla_{\theta'} \log \pi_\theta(\boldsymbol{\tau}')$ from the post-update parameter θ' into θ. Although J_{post} takes into account the functional relationship between θ' and θ, it does not take into account the pre-update sampling distribution $P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)$.

This is where $\nabla_\theta J_{\text{pre}}$ comes into play: $\nabla_\theta J^I_{\text{pre}}$ can be viewed as policy gradient update of the pre-update policy π_θ w.r.t. to the post-update return $R(\tau')$. Hence this gradient term aims to shift the pre-update sampling distribution so that higher post-update returns are achieved. However, $\nabla_\theta J^{II}_{\text{pre}}$ does not take into account the causal dependence of the post-update policy on the pre-update policy. Thus a change in θ due to $\nabla_\theta J^{II}_{\text{pre}}$ may counteract the change due to $\nabla_\theta J^{II}_{\text{post}}$. In contrast, $\nabla_\theta J^I_{\text{pre}}$ takes the dependence of the the post-update policy on the pre-update sampling distribution into account. Instead of simply weighting the gradients of the pre-update policy $\nabla_\theta \log \pi_\theta(\boldsymbol{\tau})$ with $R(\tau')$ as in $\nabla_\theta J^I_{\text{post}}$, $\nabla_\theta J^I_{\text{post}}$ weights the gradients with inner product of the pre-update and post-update policy gradients. This inner product can be written as

$$\nabla_\theta J^{\text{inner}\top} \nabla_{\theta'} J^{\text{outer}} = ||\nabla_\theta J^{\text{inner}}||_2 \cdot ||\nabla_{\theta'} J^{\text{outer}}||_2 \cdot \cos(\delta) \tag{32}$$

wherein δ denotes the angle between the the inner and outer pre-update and post-update policy gradients. Hence, $\nabla_\theta J^I_{\text{post}}$ steers the pre-update policy towards not only towards larger post-updates returns but also towards larger adaptation steps $\alpha \nabla_\theta J^{\text{inner}}$, and better alignment of pre- and post-update policy gradients. This directly optimizes for maximal improvement/adaptation for the respective task. See [8, 9] for a comparable analysis in case of domain generalization and supervised meta-learning. Note that (32) allows formulation I to perform credit assignment on the trajectory level whereas formulation II can only assign credit to entire batches of N pre-update trajectories $\tau^{1:N}$.

B Estimating the Meta-Policy Gradients

When employing formulation I for gradient-based meta-learning, we aim maximize the loss

$$J(\theta) = \mathbb{E}_{\mathcal{T}\sim\rho(\mathcal{T})} \left[\mathbb{E}_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')} \left[R(\boldsymbol{\tau}') \right] \right] \quad \text{with} \quad \theta' := \theta + \alpha \, \nabla_\theta \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)} \left[R(\boldsymbol{\tau}) \right] \tag{33}$$

by performing a form of gradient-descent on $J(\theta)$. Note that we, from now on, assume $J := J^I$ and thus omit the superscript indicating the respective meta-learning formulation. As shown in A.2 the gradient can be derived as $\nabla_\theta J(\theta) = \mathbb{E}_{(T)\sim\rho(T)}[\nabla_\theta J_{\mathcal{T}}(\theta)]$ with

$$\nabla_\theta J_{\mathcal{T}}(\theta) = E_{\boldsymbol{\tau}'\sim P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta')} \left[\nabla_{\theta'} \log P_{\mathcal{T}}(\boldsymbol{\tau}'|\theta') R(\boldsymbol{\tau}') \left(I + \alpha \nabla^2_\theta \, \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)} \left[R(\boldsymbol{\tau}) \right] \right) \right] \tag{34}$$

where $\nabla^2_\theta J_{\text{inner}}(\theta) := \nabla^2_\theta \, \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)} \left[R(\boldsymbol{\tau}) \right]$ denotes hessian of the inner adaptation objective w.r.t. θ. This section concerns the question of how to properly estimate this hessian.

B.1 A decomposition of the hessian

Estimating the the hessian of the reinforcement learning objective has been discussed in [6] and [2] with focus on second order policy gradient methods. In the infinite horizon MDP case, [2] derive a decomposition of the hessian. In the following, we extend their finding to the finite horizon case.

Proposition. The hessian of the RL objective can be decomposed into four matrix terms:

$$\nabla^2_\theta J_{\text{inner}}(\theta) = \mathcal{H}_1 + \mathcal{H}_2 + \mathcal{H}_{12} + \mathcal{H}_{12}^\top \tag{35}$$

where

$$\mathcal{H}_1 = \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t, \boldsymbol{s}_t) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t, \boldsymbol{s}_t)^\top \left(\sum_{t'=t}^{H-1} r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})\right)\right] \tag{36}$$

$$\mathcal{H}_2 = \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1} \nabla^2_\theta \log \pi_\theta(\boldsymbol{a}_t, \boldsymbol{s}_t) \left(\sum_{t'=t}^{H-1} r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})\right)\right] \tag{37}$$

$$\mathcal{H}_{12} = \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t, \boldsymbol{s}_t) \nabla_\theta Q_t^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t)^\top\right] \tag{38}$$

Here $Q_t^{\pi_\theta}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \mathbb{E}_{\boldsymbol{\tau}^{t+1:H-1}\sim P_\mathcal{T}(\cdot|\theta)}\left[\sum_{t'=t}^{H-1} r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|s_t, a_t\right]$ denotes the expected state-action value function under policy π_θ at time t.

Proof. As derived in (18), the hessian of $J_{\text{inner}}(\theta)$ follows as:

$$\nabla^2_\theta J_{\text{inner}} = \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\left(\nabla^2_\theta \log \pi_\theta(\boldsymbol{\tau}) + \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}) \nabla_\theta \log \pi_\theta(\boldsymbol{\tau})^\top\right)\right] \tag{39}$$

$$= \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t} \nabla^2_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'})\right) r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{40}$$

$$+ \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'})\right)\left(\sum_{t'=0}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'})\right)^\top r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{41}$$

$$= \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1} \nabla^2_\theta \log \pi_\theta(\boldsymbol{a}_t, \boldsymbol{s}_t)\left(\sum_{t'=t}^{H-1} r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})\right)\right] \tag{42}$$

$$+ \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\sum_{h=0}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'}) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_h, \boldsymbol{s}_h)^\top\right) r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{43}$$

The term in (42) is equal to $\mathcal{H}_2$. We continue by showing that the remaining term in (43) is equivalent to $\mathcal{H}_1 + \mathcal{H}_{12} + \mathcal{H}_{12}^\top$. For that, we split the inner double sum in (43) into three components:

$$\mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\sum_{h=0}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'}) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_h, \boldsymbol{s}_h)^\top\right) r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{44}$$

$$= \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'}) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'})^\top\right) r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{45}$$

$$+ \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\sum_{h=0}^{t'-1} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'}) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_h, \boldsymbol{s}_h)^\top\right) r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{46}$$

$$+ \mathbb{E}_{\boldsymbol{\tau}\sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\sum_{h=t'+1}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}, \boldsymbol{s}_{t'}) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_h, \boldsymbol{s}_h)^\top\right) r(\boldsymbol{s}_t, \boldsymbol{a}_t)\right] \tag{47}$$

By changing the backward looking summation over outer products into a forward looking summation of rewards, (45) can be shown to be equal to $\mathcal{H}_1$:

$$\mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'},\boldsymbol{s}_{t'})\nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'},\boldsymbol{s}_{t'})^\top\right) r(\boldsymbol{s}_t,\boldsymbol{a}_t)\right] \tag{48}$$

$$= \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(\boldsymbol{a}_t,\boldsymbol{s}_t)\nabla_\theta \log \pi_\theta(\boldsymbol{a}_t,\boldsymbol{s}_t)^\top\left(\sum_{t'=t}^{H-1} r(\boldsymbol{s}_{t'},\boldsymbol{a}_{t'})\right)\right] = \mathcal{H}_1 \tag{49}$$

By simply exchanging the summation indices t' and h in (47) it is straightforward to show that (47) is the transpose of (46). Hence it is sufficient to show that (46) is equivalent to $\mathcal{H}_{12}$. However, instead of following the direction of the previous proof we will now start with the definition of $\mathcal{H}_{12}$ and derive the expression in (46).

$$\mathcal{H}_{12} = \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(\boldsymbol{a}_t,\boldsymbol{s}_t)\nabla_\theta Q_t^{\pi_\theta}(\boldsymbol{s}_t,\boldsymbol{a}_t)^\top\right] \tag{50}$$

$$\tag{51}$$

The gradient of $Q_t^{\pi_\theta}$ can be expressed recursively:

$$\nabla_\theta Q_t^{\pi_\theta}(\boldsymbol{s}_t,\boldsymbol{a}_t) = \nabla_\theta \mathbb{E}_{\substack{\boldsymbol{s}_{t+1}\\ \boldsymbol{a}_{t+1}}}\left[Q_{t+1}^{\pi_\theta}(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1})\right] \tag{52}$$

$$= \mathbb{E}_{\substack{\boldsymbol{s}_{t+1}\\ \boldsymbol{a}_{t+1}}}\left[\nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t+1},\boldsymbol{s}_{t+1})Q_{t+1}^{\pi_\theta}(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1}) + \nabla_\theta Q_{t+1}^{\pi_\theta}(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1})\right] \tag{53}$$

By induction, it follows that

$$\nabla_\theta Q_t^{\pi_\theta}(\boldsymbol{s}_t,\boldsymbol{a}_t) = \mathbb{E}_{\boldsymbol{\tau}^{t+1:H-1}\sim P_{\mathcal{T}}(\cdot|\theta)}\left[\sum_{t'=t+1}^{H-1}\nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'},\boldsymbol{s}_{t'})\left(\sum_{h=t'}^{H-1} r(\boldsymbol{s}_h,\boldsymbol{a}_h)\right)\right] \tag{54}$$

When inserting (54) into (50) and swapping the summation, we are able to show that $\mathcal{H}_{12}$ is equivalent to (46).

$$\mathcal{H}_{12} = \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\sum_{t'=t+1}^{H-1}\nabla_\theta \log \pi_\theta(\boldsymbol{a}_t,\boldsymbol{s}_t)\nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'},\boldsymbol{s}_{t'})^\top\left(\sum_{h=t'}^{H-1} r(\boldsymbol{s}_h,\boldsymbol{a}_h)\right)\right] \tag{55}$$

$$= \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\left(\sum_{t'=0}^{t}\sum_{h=0}^{t'-1}\nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'},\boldsymbol{s}_{t'})\nabla_\theta \log \pi_\theta(\boldsymbol{a}_h,\boldsymbol{s}_h)^\top\right) r(\boldsymbol{s}_t,\boldsymbol{a}_t)\right] \tag{56}$$

This concludes the proof that the hessian of the expected sum of rewards under policy π_θ and an MDP with finite time horizon H can be decomposed into $\mathcal{H}_1 + \mathcal{H}_2 + \mathcal{H}_{12} + \mathcal{H}_{12}^\top$.

□

B.2 Bias and variance of the curvature estimate

As shown in the previous section, $\nabla_\theta^2 J^{\text{DICE}}$ provides an unbiased estimate of the hessian of the inner objective $J_{\text{inner}} = \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[R(\boldsymbol{\tau})\right]$. However, recall the DICE objective involves a product of importance weights along the trajectory.

$$J^{\text{DICE}} = \sum_{t=0}^{H-1}\left(\prod_{t'=0}^{t}\frac{\pi_\theta(a_{t'}|s_{t'})}{\perp(\pi_\theta(a_{t'}|s_{t'}))}\right) r(s_t,a_t) \tag{57}$$

This outer product of sums can be decomposed into three terms $\mathcal{H}_1 + \mathcal{H}_{12} + \mathcal{H}_{12}^\top$ (see Appendix B.1). As noted by [6], $\mathcal{H}_{12} + \mathcal{H}_{12}^\top$ is particularly difficult to estimate. In section 4.2 we empirically show that the high variance curvature estimates obtained with the DICE objective require large batch sizes and impede sample efficient learning.

In the following we develop a low variance curvature (LVC) estimator J^{LVC} which matches J^{DICE} at the gradient level and yields lower-variance estimates of the hessian by neglecting $\mathcal{H}_{12} + \mathcal{H}_{12}^\top$. Before

formally introducing J^{LVC}, we motivate such estimator starting with the policy gradient estimate that was originally derived in [13], followed by marginalizing the trajectory level distribution $P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)$ over states s_t and actions a_t. Note that we omit reward baselines for notational simplicity.

$$\nabla_\theta J_{\text{inner}} = \mathbb{E}_{\boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta)}\left[\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right)\right] \tag{58}$$

$$= \sum_{t=0}^{H-1}\mathbb{E}_{\substack{\boldsymbol{s}_t\sim p_t^{\pi_\theta}(s_t)\\ \boldsymbol{a}_t\sim\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right)\right] \tag{59}$$

In that, $p_t^{\pi_\theta}(s_t)$ denotes the state visitation frequency at time step t, i.e. the probability density of being in s_t after t steps under the policy π_θ. In the general case $p_t^{\pi_\theta}(s_t)$ is intractable but depends on the policy parameter θ. We make the simplifying assumption that $p_t^{\pi_\theta}(s_t)$ is fixed in a local region of θ. Since we make this assumption at the gradient level, this corresponds to a 1st order Taylor expansion of $p_t^{\pi_\theta}(s_t)$ in θ. Note that this assumption is also used in the Monotonic Policy Improvement Theory [7, 10]. Based on this condition, the hessian follows as derivative of (59) whereby a "stop_gradient" expression around the state visitation frequency $p_t^{\pi_\theta}(s_t)$ resembles the 1st order Taylor approximation:

$$\mathbb{E}_{\boldsymbol{\tau}}\left[\nabla_\theta^2 J^{\text{LVC}}\right] = \nabla_\theta \sum_{t=0}^{H-1}\mathbb{E}_{\substack{\boldsymbol{s}_t\sim\perp(p_t^{\pi_\theta}(s_t))\\ \boldsymbol{a}_t\sim\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right)\right] \tag{60}$$

$$= \sum_{t=0}^{H-1}\mathbb{E}_{\substack{\boldsymbol{s}_t\sim\perp(p_t^{\pi_\theta}(s_t))\\ \boldsymbol{a}_t\sim\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)}}\Bigg[\nabla_\theta \log \pi_\theta(a_t|s_t)\nabla_\theta \log \pi_\theta(a_t|s_t)^\top\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right) \tag{61}$$

$$+ \nabla_\theta^2 \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right)\Bigg] \tag{62}$$

Since the expectation in (60) is intractable it must be evaluated by a monte carlo estimate. However, simply replacing the expectation with an average of samples trajectories induces a wrong hessian that does not correspond to (62) since outer product of log-gradients would be missing when differentiated. To ensure that automatic differentiation still yields the correct hessian, we add a "dry" importance weight comparable to DICE:

$$\nabla_\theta J^{\text{LVC}} = \sum_{t=0}^{H-1}\frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))}\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right) \quad \boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \tag{63}$$

When integrated this resembles the LVC "surrogate" objective J^{LVC}.

$$J^{\text{LVC}} = \sum_{t=0}^{H-1}\frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))}\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right) \quad \boldsymbol{\tau}\sim P_{\mathcal{T}}(\boldsymbol{\tau}|\theta) \tag{64}$$

The gradients of J^{LVC} match $\nabla_\theta J^{\text{DICE}}$ and resemble an unbiased policy gradient estimate:

$$\nabla_\theta J^{\text{LVC}} = \sum_{t=0}^{H-1}\frac{\nabla_\theta\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))}\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right) \tag{65}$$

$$= \sum_{t=0}^{H-1}\frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))}\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right) \tag{66}$$

$$\rightarrow \sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'})\right) \tag{67}$$

The respective Hessian can be obtained by differentiating (66):

$$\nabla^2_\theta J^{\text{LVC}} = \nabla_\theta \sum_{t=0}^{H-1} \frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))} \nabla_\theta \log \pi_\theta(a_t|s_t) \left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'}) \right) \tag{68}$$

$$= \sum_{t=0}^{H-1} \frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))} \nabla_\theta \log \pi_\theta(a_t|s_t) \nabla_\theta \log \pi_\theta(a_t|s_t)^\top \left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'}) \right) \tag{69}$$

$$+ \frac{\pi_\theta(a_t|s_t)}{\perp(\pi_\theta(a_t|s_t))} \nabla^2_\theta \log \pi_\theta(a_t|s_t) \left(\sum_{t'=t}^{H-1} r(s_{t'}, a_{t'}) \right) \tag{70}$$

$$\rightarrow \sum_{t=0}^{H-1} \left(\sum_{t'=0}^{t} \nabla_\theta \log \pi_\theta(a_{t'}|s_{t'}) \nabla_\theta \log \pi_\theta(a_t|s_t)^\top \right) r(s_t, a_t) \tag{71}$$

$$+ \left(\sum_{t'=0}^{t} \nabla^2_\theta \log \pi_\theta(a_{t'}|s_{t'}) \right) r(s_t, a_t) \tag{72}$$

In expectation $\nabla^2_\theta J^{\text{LVC}}$ is equivalent to $\mathcal{H}_1 + \mathcal{H}_2$:

$$\mathbb{E}_{\boldsymbol{\tau} \sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)} \left[J^{\text{LVC}} \right] = \mathbb{E}_{\boldsymbol{\tau} \sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)} \left[\sum_{t=0}^{H-1} \left(\sum_{t'=0}^{t} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_{t'}|\boldsymbol{s}_{t'}) \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)^\top \right) r(\boldsymbol{s}_t, \boldsymbol{a}_t) \right] \tag{73}$$

$$+ \mathbb{E}_{\boldsymbol{\tau} \sim P_\mathcal{T}(\boldsymbol{\tau}|\theta)} \left[\sum_{t=0}^{H-1} \left(\sum_{t'=0}^{t} \nabla^2_\theta \log \pi_\theta(\boldsymbol{a}_{t'}|\boldsymbol{s}_{t'}) \right) r(\boldsymbol{s}_t, \boldsymbol{a}_t) \right] = \mathcal{H}_1 + \mathcal{H}_2 \tag{74}$$

The Hessian $\nabla^2_\theta J^{\text{LVC}}$ no longer provides an unbiased estimate of $\nabla^2_\theta J_{\text{inner}}$ since neglects the matrix term $\mathcal{H}_{12} + \mathcal{H}_{12}^\top$. This approximation is based on the assumption that the state visitation distribution is locally unaffected by marginal changes in θ and leads to a substantial reduction of variance in the hessian estimate. [6] show that under certain conditions (i.e. infinite horizon MDP, sufficiently rich policy parameterisation) the term $\mathcal{H}_{12} + \mathcal{H}_{12}^\top$ vanishes around a local optimum θ^*. Given that the conditions hold, this implies that $\mathbb{E}_{\boldsymbol{\tau}}[\nabla^2_\theta J^{\text{LVC}}] \rightarrow \mathbb{E}_{\boldsymbol{\tau}}[\nabla^2_\theta J^{\text{DICE}}]$ as $\theta \rightarrow \theta^*$, i.e. the bias of the LCV estimator becomes negligible close to the local optimum.

C Proximal Meta-Policy Search

Algorithm 1 Proximal Meta-Policy Search (ProMP)

Require: Task distribution ρ, step sizes α, β, KL-penalty coefficient η, clipping range ϵ

Randomly initialize θ
while θ not converged **do**
  Sample batch of tasks $\mathcal{T}_i \sim \rho(\mathcal{T})$
  for step $n = 0, ..., N-1$ **do**
    if $n = 0$ **then**
      Set $\theta_o \leftarrow \theta$
      for all $\mathcal{T}_i \sim \rho(\mathcal{T})$ **do**
        Sample pre-update trajectories $\mathcal{D}_i = \{\tau_i\}$ from $\mathcal{T}_i$ using π_θ
        Compute adapted parameters $\theta'_{o,i} \leftarrow \theta + \alpha \nabla_\theta J^{LR}_{\mathcal{T}_i}(\theta)$ with $\mathcal{D}_i = \{\tau_i\}$
        Sample post-update trajectories $\mathcal{D'}_i = \{\tau'_i\}$ from $\mathcal{T}_i$ using $\pi_{\theta'_{o,i}}$
    Update $\theta \leftarrow \theta + \beta \sum_{\mathcal{T}_i} \nabla_\theta J^{\text{ProMP}}_{\mathcal{T}_i}(\theta)$ using each $\mathcal{D}'_i = \{\tau'_i\}$