CURRICULUM FOR GRADIENT-BASED META-LEARNERS

Bhairav Mehta, Tristan Deleu, Sharath Chandra Raparthy
Mila, Université de Montréal

Christopher J. Pal
Mila, Polytechnique Montréal, ElementAI

Liam Paull
Mila, Université de Montréal

ABSTRACT

Gradient-based meta-learners such as Model-Agnostic Meta-Learning (MAML) have shown strong few-shot performance in supervised and reinforcement learning settings. However, specifically in the case of meta-reinforcement learning (meta-RL), we can show that gradient-based meta-learners are sensitive to task distributions. With the wrong curriculum, agents suffer the effects of *meta-overfitting*, shallow adaptation, and adaptation instability. In this work, we begin by highlighting intriguing failure cases of gradient-based meta-RL and show that task distributions can wildly affect algorithmic outputs, stability, and performance. Next, to address this problem, we leverage insights from recent literature on *domain randomization*. Specifcally, we propose meta-Active Domain Randomization (meta-ADR), which learns a curriculum of tasks for gradient-based meta-RL in a similar as ADR does for sim2real transfer. We show that this approach induces more stable policies on a variety of simulated locomotion and navigation tasks. We assess in- and out-of-distribution generalization and find that the learned task distributions, even in an unstructured task space, greatly improve the adaptation performance of MAML.

Meta-learning concerns building models or agents that can learn how to adapt quickly to new tasks from datasets which are orders of magnitudes smaller than their standard supervised learning counterparts. Put differently, meta-learning concerns learning *how to learn*, rather than simply maximizing performance on a single task or dataset. Gradient-based meta-learning has seen a surge of interest, with the foremost algorithm being Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017a). Gradient-based meta-learners are fully trainable via gradient descent, and have shown strong performance on various supervised and reinforcement learning tasks.

The focus of this work is on an understudied hyperparameter within the gradient-based meta-learning framework: $p(\tau)$. Symbolizing the distribution of tasks, this distribution is assumed given in MAML, and is used to sample tasks for meta-training of the MAML agent. In supervised learning, this quantity is relatively well defined, as we have often have a large dataset for a task such as image classification [1]. As a result, $p(\tau)$ is built from random minibatches sampled from this distribution.

However, in the meta-reinforcement learning setting, this task distribution is poorly defined, and is often handcrafted by a human experimenter with the target task in mind. While the task samples themselves are pulled randomly from a range or distribution (i.e a locomoter asked to achieve a target velocity $v_t \sim U(-3, 3)$), the distribution *itself* needs to be specified. In practice, the distribution $p(\tau)$ turns out to be an extremely sensitive hyperparameter in Meta-RL: too ”wide” leads to underfitting, with agents unable to specialize to the given target task even with larger numbers of gradient steps; too ”narrow”, and we see poor generalization and adaption to even slightly out-of-distribution environments.

[1]Even in regression, a function such as a sinusoid is often provided.

Even worse, randomly sampling (as is often the case) from $p(\tau)$ can allow for sampling of tasks that can cause interference and optimization difficulties, especially when tasks are qualitatively different (due to difficulty or task definitions being changed too much by the physical parameters that are varied).

This phenomena, called *meta-overfitting* (or meta-underfitting, in the former, "wide" case), is not new to recent deep reinforcement learning problem settings. Domain randomization (Tobin et al., 2017), a popular *sim2real* transfer method, faces many of the same issues when learning robotic policies purely in simulation. Here, we will show that meta-reinforcement learning has the analogous issues regarding generalization and we will describe the repurposing of a recent algorithm called *Active Domain Randomization* (Mehta et al., 2019), which aims to learn a curriculum of tasks in unstructured task spaces. The incorporation of a learned curriculum leads to stronger generalization performance and more robust optimization. Our results highlight the need for continued work in analysis of the effect of task distributions on meta-RL performance and underscoring the potential for curriculum learning techniques.

1 Background

We refer readers to the appendix for a background on the standard reinforcement learning (Sutton & Barto, 2018), meta-learning, meta-reinforcement learning Finn et al. (2017b), and curriculum learning problem settings.

1.1 Active Domain Randomization

Active Domain Randomization (ADR) (Mehta et al., 2019) builds upon the framework of Domain Randomization (Tobin et al., 2017) by learning an active policy that proposes a curriculum of tasks to train an inner-loop, black-box agent. ADR, mostly used in the zero-shot learning scenario of *simulation-to-real transfer*, uses Stein Variational Policy Gradient (SVPG) (Liu et al., 2017) to learn a set of parameterized particles, $\{\mu_{\phi_i}\}_{i=1}^N$ that proposed randomized environments that are subsequently used to train the agent. The full update can be written as

$$\mu_{\phi_i} \leftarrow \mu_{\phi_i} + \frac{\epsilon}{N}\Sigma_{j=1}^{N}[\nabla_{\mu_{\phi_j}} J(\mu_{\phi_j})k(\mu_{\phi_i}, \mu_{\phi_j}) + \alpha\nabla_{\mu_{\phi_j}} k(\mu_{\phi_i}, \mu_{\phi_j})], \tag{1}$$

where $J(\mu_{\phi_i})$ denotes the sampled return from particle i, and the learning rate ϵ and temperature α are hyperparameters.

SVPG benefits from both the Maximum-Entropy RL framework (Ziebart, 2010; Haarnoja et al., 2018) and a kernel term that repulses similar policies to encourage particle, and therefore task, diversity. This allows SVPG to hone in on regions of high reward while maintaining variety, which allows ADR to outperform many existing methods in terms of performance and generalization on zero-shot learning and robotic benchmarks. ADR uses a discriminator to distinguish between trajectories generated in a proposed randomized environment and those generated by the same policy in a default, reference environment. Intuitively, ADR is optimized to find environments where the same policy produces different behavior in the two types of environments, signalling a probable weakness in the policy when evaluated on those types of randomized environments.

2 Motivation

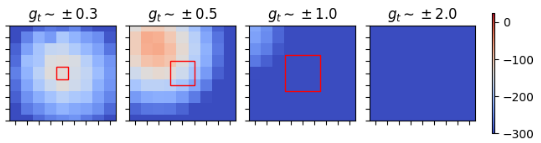
We begin with a simple question:

Does the meta-training distribution in meta-RL really matter?

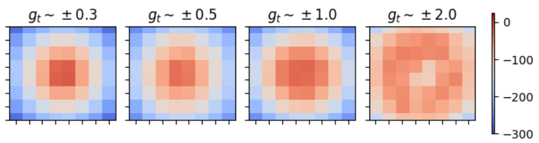
To answer this question, we run a standard meta-reinforcement learning benchmark, *2D-Navigation-Dense*. While *2D-Navigation* comes in both sparse-reward (i.e rewards are nonzero only when reaching the goal) and dense-reward (i.e rewards given at each timestep proportional to the Euclidean distance between the goal and the current position) variants, we focus on the later as the former requires many more gradient updates in order to solve.

We take the configuration from (Finn et al., 2017a) and simply change *the distribution* from which the 2D goal is *uniformly* sampled from. We then show generalization results of the final, meta-learned initial policy *after a single gradient step*. We then reset the policy to the same, final initialization, and track the generalization of the one-step adaptation across a wide range of target goals.

In *2D-Navigation-Dense*, the training distribution prescribes goals where each coordinate is traditionally sampled between $[-0.5, 0.5]$ (the second plot in Figure 1a) with the agent always beginning at $[0, 0]$. We then evaluate each goal in the grid between $[-2, 2]$ at 0.5 intervals, allowing us to test both in- and out-of-distribution generalization.



(a) Generalization with Standard MAML



(b) Generalization with Meta-ADR

Figure 1: Generalization in 2D-Navigation, redder is higher reward.

We see from Figure 1a an interesting phenomenon, particularly as the training environment shifts away from the one which samples goal coordinates $g_t \sim [-0.5, 0.5]$. While the standard environment from (Finn et al., 2017a) generalizes reasonably well, shifting the training distribution even slightly ruins generalization of the adapted policy. What's more, when shown the entire test distribution, MAML fails to generalize to it. We see that on even the simplest environments, the meta-training task distribution seems to have a profound effect, motivating the need for dedicating more attention towards selecting the task distribution $p(\tau)$.

Upon further inspection, we find that shifting the meta-training distribution destabilizes MAML, leading to poor performance when averaged. The first environment, where $g_t \sim [-0.3, 0.3]$ has three out of five random seeds that converge, with the latter two, $g_t \sim [-1.0, 1.0]$ and $g_t \sim [-2.0, 2.0]$, have two and one seeds that converge respectively. The original task distribution sees convergence in all five random seeds tested, hinting at a difference in stability due to the goals, and therefore task distribution, that each agent sees.

3 METHOD

In this work, we relax the assumption of (given) uniformly-sampled, random tasks, and instead optimize a curriculum over the task distribution $p(\tau)$:

$$\arg\min_{\tau_i \sim p(\tau)} \min_{\theta} \Sigma_{\tau_i} \mathcal{L}_{\tau_i}(f_{\theta_i^{'}}) \tag{2}$$

where $\theta_i^{'}$ are the updated parameters after a single meta-gradient update.

While curriculum learning has had success in scenarios where task-spaces are structured, learning curricula in unstructured task-spaces, where an intuitive scale of *difficulty* might be lacking, is an understudied topic. However, learning such curricula has seen a surge of interest in the problem of *simulation transfer* in robotics, where policies trained in simulation are transferred zero-shot (no fine-tuning) for use on real hardware. Using a method called domain randomization (Tobin et al., 2017), several recent methods (OpenAI et al., 2019; Mozifian et al., 2019) propose how to learn a curriculum of *randomizations* - which randomized environments would be most useful to show the learning agent in order to make progress on the held-out target task: the real robot.

In the meta-RL setting, the learned curriculum would be over the space of tasks. For example, in *2D-Navigation-Dense*, this would be where goals are sampled, or in *HalfCheetahVelocity*, another popular meta-RL benchmark, the goal velocity the locomotor must achieve.

As learning the curriculum is often treated as a reinforcement learning problem, it requires a reward in order to calculate policy gradients. While many of the methods from the domain randomization literature use proxies such as completion rates or average reward, the optimization scheme depends on the reward function of the *task*. In meta-learning, optimization and reward maximization on a *single* task is not the goal, and such an approach may lead to counter-intuitive results.

A more natural fit in the meta-learning scenario would be to somehow use the *qualitative difference* between the pre- and post-adaptation trajectories. Like a good teacher with a struggling student, the curriculum could shift towards where the meta-learner needs help. For example, tasks in which *negative adaptation* (Deleu & Bengio, 2018) occurs, or where the return from a pre-adapted agent is higher the post-adapted agent, would be prime tasks to focus on for training.

To this end, we modify *Active Domain Randomization* (ADR) to calculate such a score between the two types of trajectories. Rather than using a reference environment as in ADR, we ask a discriminator to label the source of the pre- and post-adaptation trajectories. If a particular task generates trajectories that can be distinguished by the discriminator after adaptation, we focus more heavily on these tasks by providing the high-level optimizer, parameterized by Stein Variational Policy Gradient (Eq. 1), a higher reward.

Concretely, we provide the particles the reward $r_i = \log(f_\psi(y|D_i))$, where discriminator f_ψ produces a boolean prediction of whether the trajectory D_i is a pre-adaptation ($y = 0$) or post-adaptation ($y = 1$) trajectory. Meta-ADR learns a curriculum in this unstructured task space without relying on the notion of task performance or reward functions. Note that Meta-ADR runs the original MAML algorithm as a subroutine, but in fact Meta-ADR can run any meta-learning subroutine (i.e Reptile (Nichol et al., 2018), PEARL (Rakelly et al., 2019), or First-Order MAML). In this work, we abstract away the meta-learning subroutine, focusing instead on the effect of task distributions on the learner's generalization capabilities. An advantage of Meta-ADR over the ADR original formulation is that unlike ADR, Meta-ADR requires no additional rollouts, using the rollouts already required by gradient-based meta-reinforcement learners to optimize the curriculum.

We show results of Meta-ADR in *2D-Navigation-Dense* in Figure 1b. The curriculum learning aspect of the approach stabilizes generalization, leading to plots that show the true fast-adaptation qualities of MAML. To test this in higher dimensions, we turn to *Humanoid-Direc-2D*. In the standard variant of *Humanoid-Direc-2D*, a locomoter is tasked with running in a particular direction, sampled from $[0, 2\pi]$. This task makes no distinction regarding target velocity, but rather calculates the reward based on the agent's heading and other control costs. *Humanoid-Direc-2D* is a high-dimensional control problem, and represents one of the most difficult benchmarks in use in Meta-RL literature today.

To see the effects of Meta-ADR, In this task, we shift the distribution from $[0, 2\pi]$ to subsets of this range, subsequently training and evaluating MAML agents across the entire range of tasks between $[0, 2\pi]$. Again, we compare agents trained with the standard uniformly-random sampled task distribution against those trained with a learned curriculum using Meta-ADR.

When studying the generalization capabilities on this difficult continuous control task, we are particularly interested in *symmetric* versions of the task; for example, tasks that sample the right and left semi-circles of the task space. We repeat this experiment with many variants of this symmetric task description, and report representative results due to space in Figure 2.

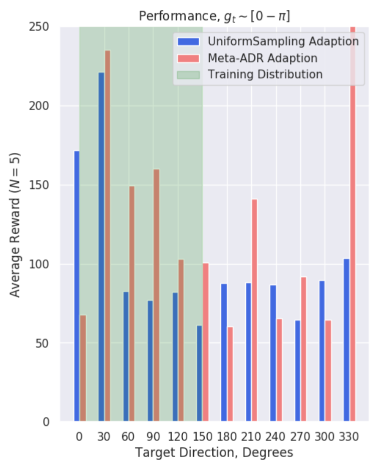
REFERENCES

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553380. URL http://doi.acm.org/10.1145/1553374.1553380.

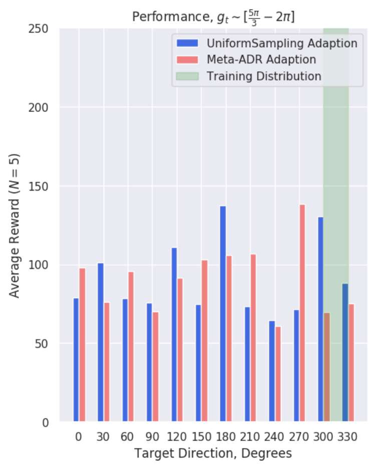
Tristan Deleu and Yoshua Bengio. The effects of negative adaptation in Model-Agnostic Meta-Learning. *CoRR*, abs/1812.02159, 2018. URL http://arxiv.org/abs/1812.02159.

Tristan Deleu and Hosseini Seyedarian Guiroy, Simon. Reinforcement Learning with Model-Agnostic Meta-Learning in PyTorch, 2018. URL https://github.com/tristandeleu/pytorch-maml-rl/.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL2: Fast Reinforcement Learning via Slow Reinforcement Learning. 2016. URL http://arxiv.org/abs/1611.02779.



(a) Generalization with Meta-ADR



(b) Generalization with Meta-ADR

Figure 2: In complex, high-dimensional environments, training task distributions can wildly vary performance. Even in the Humanoid Directional task, Meta-ADR allows MAML to generalize across the range, although it too is affected in terms of total return when compared to the same algorithm trained with ”good” task distributions.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *International Conference on Machine Learning*, 2017a. URL http://arxiv.org/abs/1703.03400.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-Shot Visual Imitation Learning via Meta-Learning. *Conference on Robot Learning*, 2017b. URL https://arxiv.org/abs/1709.04905.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning, 2017.

Alex Graves, Marc G. Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks, 2017.

Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies, 2018.

Swaminathan Gurumurthy, Sumit Kumar, and Katia Sycara. Mame : Model-agnostic meta-exploration, 2019.

Tuomas Haarnoja, Vitchyr Pong, Aurick Zhou, Murtaza Dalal, Pieter Abbeel, and Sergey Levine. Composable deep reinforcement learning for robotic manipulation. *arXiv preprint arXiv:1803.06773*, 2018.

Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, S. M. Ali Eslami, Martin Riedmiller, and David Silver. Emergence of locomotion behaviours in rich environments, 2017.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese Neural Networks for One-shot Image Recognition. *International Conference on Machine Learning*, 2015.

Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. Stein variational policy gradient, 2017.

Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active domain randomization. *CoRR*, abs/1904.04762, 2019. URL http://arxiv.org/abs/1904.04762.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A Simple Neural Attentive Meta-Learner. *International Conference on Learning Representations*, 2018. URL https://arxiv.org/pdf/1707.03141.pdf.

Melissa Mozifian, Juan Camilo Gamboa Higuera, David Meger, and Gregory Dudek. Learning domain randomization distributions for transfer of locomotion policies. *CoRR*, abs/1906.00410, 2019. URL http://arxiv.org/abs/1906.00410.

Tsendsuren Munkhdalai and Hong Yu. Meta Networks. *International Conference on Machine Learning*, 2017. URL https://arxiv.org/abs/1703.00837.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to Adapt in Dynamic, Real-World Environments Through Meta-Reinforcement Learning. *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1803.11347.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018.

OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jadwiga Tworek, Peter Welinder, Lilian Weng, Qi-Ming Yuan, Wojciech Zaremba, and Lefei Zhang. Solving rubik's cube with a robot hand. *ArXiv*, abs/1910.07113, 2019.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning, 2017.

Vitchyr H. Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning, 2019.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables. *International Conference on Machine Learning*, 2019. URL https://arxiv.org/abs/1903.08254.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. *International Conference on Learning Representations*, 2017.

Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. Promp: Proximal meta-policy search, 2018.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-Learning with Memory-Augmented Neural Networks. *International Conference on Machine Learning*, 2016. URL https://arxiv.org/abs/1605.06065.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust Region Policy Optimization, 2015. URL https://arxiv.org/abs/1502.05477.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical Networks for Few-shot Learning. *Conference on Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1703.05175.

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, 2018.

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

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Brian MacWhinney, and Chris Dyer. Learning the curriculum with Bayesian optimization for task-specific word representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 130–139, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1013. URL https://www.aclweb.org/anthology/P16-1013.

Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. *Conference on Neural Information Processing Systems*, 2016. URL http://arxiv.org/abs/1606.04080.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. 2016.

Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O. Stanley. Paired open-ended trailblazer (poet): Endlessly generating increasingly complex and diverse learning environments and their solutions, 2019.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine Learning*, 1992.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning, 2019.

Brian D Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, CMU, 2010.

A Appendix

B Extra Background

B.1 Reinforcement Learning

We consider a reinforcement learning setting where a task τ is defined as a Markov Decision Process (MDP), a tuple (S, A, T, R, γ) where S is the state space, A is the action space, T is transition function $T : S \times A \to S$, R is a reward function and γ is a *discount factor* which lives within $(0, 1)$. The goal of reinforcement learning is to learn a function π parameterized by θ in such a way that it maximizes the expected total discounted reward.

B.2 Meta-Learning

Most deep learning models are built to solve only one task, and often lack the ability to generalize and quickly adapt to solve a new set of tasks. Meta-learning involves learning a *learning algorithm* which can adapt quickly rather than learning from scratch. Several methods have been proposed, treating the learning algorithm as a recurrent model capable of remembering past experience (Santoro et al., 2016; Munkhdalai & Yu, 2017; Mishra et al., 2018), as a non-parametric model (Koch et al., 2015; Vinyals et al., 2016; Snell et al., 2017), or as an optimization problem (Ravi & Larochelle, 2017; Finn et al., 2017a). In this paper, we focus on a popular version of a gradient-based meta-learning algorithm called Model Agnostic Meta Learning (MAML; (Finn et al., 2017a)).

B.3 Gradient-based Meta-Learning

The main idea in MAML is to find a good parameter initialization such that the model can adapt to a new task, τ, quickly. Formally, given a distribution of tasks $p(\tau)$ and a loss function $\mathcal{L}_\tau$ corresponding to each task, the aim is to find parameters θ such that the model f_θ can adapt to new tasks with one or few gradient steps. For example, in the case of a single gradient step, the parameters θ'_τ adapted to the task τ are

$$\theta'_\tau = \theta - \alpha \nabla_\theta \mathcal{L}_\tau(\mathcal{D}_{\text{train}}, f_\theta), \tag{3}$$

where the loss is evaluated on a (typically small) dataset $\mathcal{D}_{\text{train}}$ of training examples from task τ. In order to find a good initial value of the parameters θ, the objective function being optimized in MAML is written as

$$\min_\theta \sum_{\tau_i} \mathcal{L}_{\tau_i}(\mathcal{D}_{\text{test}}, f_{\theta'_{\tau_i}}), \tag{4}$$

where it evaluates the performance in generalization on some test examples $\mathcal{D}_{\text{test}}$ for task τ. The meta objective function is optimized by gradient descent where the parameters are updated according to

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\tau_i} \mathcal{L}_{\tau_i}(\mathcal{D}_{\text{test}}, f_{\theta'_{\tau_i}}), \tag{5}$$

where α and β are step sizes.

B.4 Meta-Reinforcement Learning

In addition to few-shot supervised learning problems, where the number of training examples is small, meta-learning has also been successfully applied to reinforcement learning problems. In meta-reinforcement learning, the goal is to find a policy that can quickly adapt to new environments, generally from only a few trajectories. Rakelly *et. al.* (Rakelly et al., 2019) treat this problem by conditioning the policy on a latent representation of the task, and Duan *et. al.* and Wang *et. al.* (Duan et al., 2016; Wang et al., 2016) represent the reinforcement learning algorithm as a recurrent network, inspired by the "black-box" meta-learning methods mentioned above. Some meta-learning algorithms can even be adapted to reinforcement learning with minimal changes (Finn et al., 2017a; Mishra et al., 2018). In particular, MAML has also shown some success on robotics applications (Finn et al., 2017b; Nagabandi et al., 2019). In the context of reinforcement learning, both $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$ are datasets of trajectories sampled by the policies before and after adaptation respectively. The loss function used for the adaptation is REINFORCE (Williams, 1992), and the meta objective is optimized using TRPO (Schulman et al., 2015).

B.5 Curriculum Learning

Curriculum learning (Bengio et al., 2009) focuses on a meta-optimization task, where a black-box learner is required to learn a goal task τ_g. By abstracting away the inner-loop learner, curriculum learning focuses on *what* to show the learner in order to allow it to solve the goal task. Oftentimes, curriculum learning focuses on showing the learner easier versions of the goal task - for example, in a navigation task, closer goals may be shown to the agent first, progressively getting harder until the goal task τ_g is achieved.

C Related Work

When discussing meta-reinforcement learning, to the best of our knowledge, the task distribution $p(\tau)$ has never been studied or ablated upon. As most benchmark environments and tasks in meta-RL stem from two papers ((Finn et al., 2017a; Rothfuss et al., 2018), with the task distributions being prescribed with the environments), the discussion in meta-RL papers has almost always centered around the computation of the updates (Rothfuss et al., 2018), practical improvements and approximations made to improve efficiency, or learning exploration policies with meta-learning (Stadie et al., 2018; Gurumurthy et al., 2019; Gupta et al., 2018). In this section, we briefly discuss prior work in curriculum learning that bears the most similarity to the analyses we conduct here.

Starting with the seminal curriculum learning paper (Bengio et al., 2009), many different proposals to learn an optimal *ordering* over tasks has been studied. Curriculum learning has been tackled with Bayesian Optimization (Tsvetkov et al., 2016), multi-armed bandits (Graves et al., 2017), and evolutionary strategies (Wang et al., 2019) in supervised learning and reinforcement learning settings, but here, we focus on the latter. However, in most work, the task space is almost always discrete, with a teacher agent looking to choose the best next task over a set of N premade tasks. The notion of *best* has also been explored in depth, with metrics being based on a variety of things from ground-truth accuracy or reward to adversarial gains between a teacher and student agent (Pinto et al., 2017).

However, up until recently, the notion of continuously- parameterized curriculum learning has been studied less often. Often, continuous-task curriculum learning exploits a notion of *difficulty* in the task itself. In order to get agents to hop over large gaps, it's been empirically easier to get them to jump over smaller ones first (Heess et al., 2017); likewise, in navigation domains, its been easier to show easier goals and *grow* a goal space (Pong et al., 2019), or even work backwards towards the start state in a reverse curriculum manner (Florensa et al., 2017).

While deep reinforcement learning, particularly in robotics, has a seen a large amount of curriculum learning papers in recent times (Wang et al., 2019; OpenAI et al., 2019), curriculum learning has not been extensively researched in meta-RL. This may be partly due to the naissance of the field; only recently was a large-scale, multi-task benchmark for meta-RL released (Yu et al., 2019). As we hope to show in this work, the notions of tasks, task distributions, and curricula in meta-learning are fruitful avenues of study, and can make (or break) many of the meta-learning algorithms in use today.

D Additional Results

In this section, we show the results of uniform sampling of the standard MAML agent when changing task distribution $p(\tau)$, while also benchmarking against a MAML agent trained with a learned task distribution using Meta-ADR. All hyperparameters for each task are taken from (Finn et al., 2017a; Rothfuss et al., 2018), with the exception that we take the *final* policy at the end of 200 meta-training epochs instead of the best-performing policy over 500 meta-training epochs. We use the code from (Deleu & Guiroy, 2018) to run all of our experiments. Unless otherwise noted, all experiments are run and averaged across five random seeds. All results are shown after a single gradient step during meta-test time. For each task, we artificially create a generalization range; potentially disjoint from the training distribution of target goals, velocities, headings, etc., and we evaluate each agent both in- and out-of-distribution.

Importantly, since our focus is on generalization, we **evaluate the final policy**, rather than the standard, *best-performing* policy. As MAML produces a final *initial* policy, when evaluating for generalization for meta-learning, we adapt that initial policy to each target task, and report the adaption results. In addition, in certain sections, we discuss *negative* adaption, which is simply the performance difference between the final, adapted policy and the final, initial policy. When this quantity is negative, as noted in (Deleu & Bengio, 2018), we say that the policy has *negatively* adapted to the task.

D.1 Ant-Navigation

Interestingly, on a more complex variant of the same task described in the main text, *Ant-Navigation*, the benefits of such a learned curriculum are minimized. In this task, an eight-legged locomoter is tasked with achieving goal positions sampled from a predetermined range; the standard environment samples the goal positions from a box centered at $(0, 0)$, with each coordinate sampled from $g \sim [-3, 3]$. We systematically each agent on a grid with both axes ranging between $[-7, 7]$, with a 0.5 step interval.

In Figure 3, we qualitatively see the same generalization across all training task distributions when comparing a randomly sampled task curriculum and a learned one. We hypothesize that this stability comes mainly from the control components of the reward, leading to a smoother, stabler performance across all training distributions. In addition, generalization is unaffected by the choice of distribution, pointing to differences between this task and the simpler version.

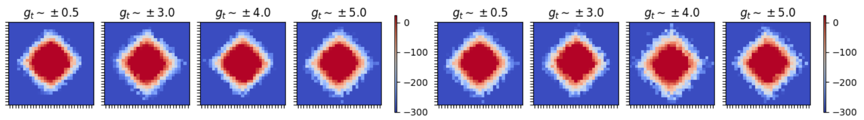


Figure 3: In the Ant-Navigation task, both uniformly sampled goals (top) and a learned curriculum of goals with Meta-ADR (bottom) are stable in performance. We attribute this to the extra components in the reward function. Redder is higher reward.

Compared to the *2D-Navigation-Dense*, *Ant-Navigation* also receives a dense reward related to its distance to target, specifically "control cost, a contact cost, a survival reward, and a penalty equal to its L1 distance to the target position." In comparison, the *2D-Navigation-Dense* task, while a simpler control problem, receives reward information only related to the Euclidean distance to the goal. Counter-intuitively, this simplicity results in *less* stable performance when uniformly sampling tasks, an ablation which we hope to study in future work.

D.2 Target Velocity Tasks

When dealing with the target velocity task, we train an ant locomoter to attain target speeds sampled from $v_t \sim [0, 3]$ (Figure 4 left, the standard variant of *AntVelocity*) and speeds sampled from $v_t \sim [0, 5]$ (Figure 4 right).

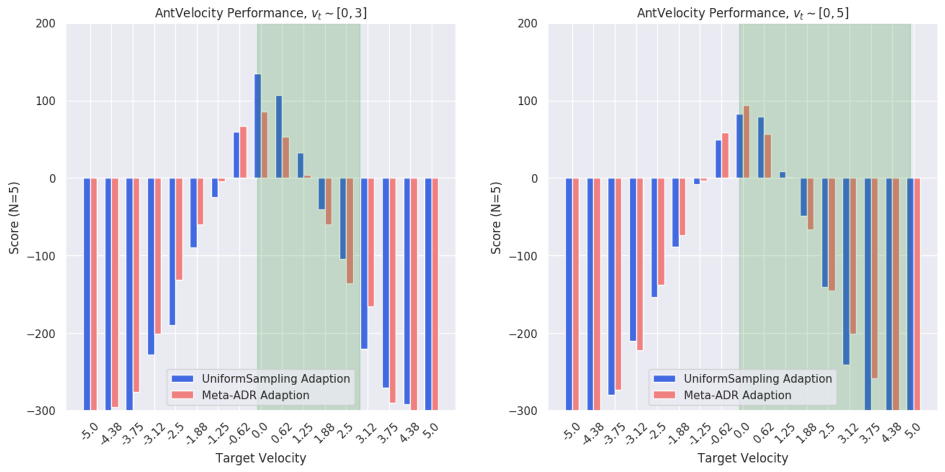


Figure 4: Ant-Velocity sees less of a benefit from curriculum, but performance is greatly affected by a correctly-calibrated task distribution (left). In a miscalibrated one (right), we see that performance from a learned curriculum is slightly more stable.

While we see that learned curricula make insignificant amounts of performance *improvement* over random sampling when shown the same task distribution, we see large differences in performance between task distributions, motivating our hypothesis that $p(\tau)$ is a crucial hyperparameter for successful meta-RL. In additon, we notice that the highest scores are attained on the velocities closer to the easiest variant of the task: a $v_t = 0$, which requires the locomoter to stand completely still.