Acceleration in Policy Optimization

Veronica Chelu [♥♠] Tom Zahavy[♣] Arthur Guez[♣] Doina Precup[♥♠♣] Sebastian Flennerhag[♣]

[♥]McGill University [♠] Mila Quebec AI Institute [♣]Google DeepMind

Abstract

We work towards a unifying paradigm for accelerating policy optimization methods in reinforcement learning (RL) through predictive and adaptive directions of (functional) policy ascent. Leveraging the connection between policy iteration and policy gradient methods, we view policy optimization algorithms as iteratively solving a sequence of surrogate objectives, local lower bounds on the original objective. We define optimism as predictive modelling of the future behavior of a policy, and hindsight adaptation as taking immediate and anticipatory corrective actions to mitigate accumulating errors from overshooting predictions or delayed responses to change. We use this shared lens to jointly express other well-known algorithms, including model-based policy improvement based on forward search, and optimistic meta-learning algorithms. We show connections with Anderson acceleration, Nesterov's accelerated gradient, extra-gradient methods, and linear extrapolation in the update rule. We analyze properties of the formulation, design an optimistic policy gradient algorithm, adaptive via meta-gradient learning, and empirically highlight several design choices pertaining to acceleration, in an illustrative task.

1 Introduction

Policy gradient (PG) methods [Williams, 1992, Sutton et al., 1999] are one of the most effective reinforcement learning (RL) algorithms [Espeholt et al., 2018, Schulman et al., 2015, 2017, Abdolmaleki et al., 2018, Hessel et al., 2021, Zahavy et al., 2020, Flennerhag et al., 2021]. These methods search for the optimal policy in a parametrized class of policies by using gradient ascent to maximize the cumulative expected reward that a policy collects when interacting with an environment. While effective, this objective poses challenges to the analysis and understanding of PG-based optimization algorithms due to its non-concavity in the policy parametrization [Agarwal et al., 2019, Mei et al., 2020b,a, 2021b].

Nevertheless, PG methods globally converge sub-linearly for smoothly parametrized softmax policy classes. This analysis relies on local linearization of the objective function in parameter space and uses small step sizes and gradient domination to control the errors introduced from the linearization [Agarwal et al., 2019, Mei et al., 2020b, 2021b, 2023]. In contrast, policy iteration (PI) linearizes the objective w.r.t. (with respect to) the functional representation of the policy [Agarwal et al., 2019, Bhandari and Russo, 2019, 2021], and converges linearly when the surrogate objective obtained from the linearization is known and can be solved in closed form.

Relying on the illuminating connections between PI and several instances of PG algorithms (including (inexact) natural policy gradients (NPG) and mirror ascent (MA)), recent works [Bhandari and Russo, 2021, Cen et al., 2022, Mei et al., 2021a, Yuan et al., 2023, Alfano and Rebeschini, 2023, Chen and Theja Maguluri, 2022] extended the above results and showed linear convergence of PG algorithms with large step sizes (adaptive or geometrically increasing). Other works showed that PG methods can achieve linear rates via entropy regularization. These guarantees cover some (approximately) closed policy classes, e.g., tabular, or log-linear—cf. Table 1 in Appendix A. More generally, in practice,

[*]Correspondence: <veronica.chelu@mail.mcgill.ca>. Work executed on internship at DeepMind.

each iteration of these PI-like algorithms is solved approximately, using a few gradient ascent update steps, in the space of policy parameters, which lacks guarantees due to non-concavity induced by non-linear transformations in the deep neural networks used to represent the policy [Agarwal et al., 2019, Abdolmaleki et al., 2018, Tomar et al., 2020, Vaswani et al., 2021].

This recent understanding about the convergence properties of policy gradient methods in RL leaves room to consider more advanced techniques. In this work, we focus on **acceleration** via *optimism*—a term we borrow from online convex optimization [Zinkevich, 2003], and is unrelated to the exploration strategy of *optimism in the face of uncertainty*. In this context, *optimism* refers to predicting future gradient directions in order to accelerate convergence (for instance, as done in Nesterov's accelerated gradients (NAG) [Nesterov, 1983, Wang and Abernethy, 2018, Wang et al., 2021], extra-gradient (EG) methods [Korpelevich, 1976], mirror-prox [Nemirovski, 2004, Juditsky et al., 2011], optimistic MD [Rakhlin and Sridharan, 2013a, Joulani et al., 2020], AO-FTRL [Rakhlin and Sridharan, 2014, Mohri and Yang, 2015], etc.).

In RL, optimistic policy iteration (OPI) [Bertsekas and Tsitsiklis, 1996, Bertsekas, 2011, Tsitsiklis, 2002] considers policy updates performed based on incomplete evaluation, with a value function estimate that gradually tracks the solution of the most recent policy evaluation problem. Non-optimistic methods, on the other hand, regard the value estimation problem as a series of isolated prediction problems and solve them by Monte Carlo or temporal difference (TD) estimation. By doing so, they ignore the potentially predictable nature of the prediction problems, and their solutions, along a policy's optimization path.

In previous work, optimism has been studied in policy optimization to mitigate oscillations [Wagner, 2014, 2013, Moskovitz et al., 2023] as well as for accelerated optimization [Cheng et al., 2018, Hao et al., 2020], resulting in sub-linear, yet unbiased convergence, cf. Table 1 in Appendix A.

In this paper, we introduce a general policy optimization framework that allows us to describe seemingly disparate algorithms as making specific choices in how they represent, or adapt optimistic gradient predictions. Central to our exposition is the idea of prognostic learning, i.e. making predictions or projections of the future behavior, performance, or state of a system, based on existing historical data (interpolation), or extending those predictions into uncharted territory by predicting beyond data (extrapolation).

In particular, we show that two classes of well-known algorithms—*meta-learning algorithms* and *model-based planning algorithms*—can be viewed as optimistic variants of vanilla policy optimization, and provide a theoretical argument for their empirical success. For example, STACX [Zahavy et al., 2020] represents an optimistic variant of Impala [Espeholt et al., 2018] and achieves a doubling of Impala's performance on the Atari-57 suite; similarly, adding further optimistic steps in BMG [Flennerhag et al., 2021] yields another doubling of the performance relative to that of STACX. In model-based RL, algorithms with extra steps of planning, e.g., the AlphaZero family of algorithms [Silver et al., 2016a, 2017], with perfect simulators, also enjoy huge success in challenging domains, e.g. chess, Go, and MuZero [Schrittwieser et al., 2019], with an *adaptive* model, achieves superhuman performance in challenging and visually complex domains.

Contributions & motivation After some background in Sec. 2, we define a simple template for accelerating policy optimization algorithms in Sec. 3. This formulation involves using proximal policy improvement methods with optimistic auto-regressive update rules, learned in hindsight, to be predictive of the post-update policy performance. We show this simple *acceleration* template based on optimism & adaptivity is a generalization of the update rule of proximal policy optimization algorithms, where the inductive bias is *fixed*, and does not change with past experience. We use the introduced generalization to show that a *learned* update rule that can form other inductive biases, that can accelerate convergence.

We use the introduced formulation to highlight the commonalities among several algorithms, expressed in this formalism in Sec. 3, including model-based policy optimization algorithms relying on run-time forward search (e.g. Silver et al. [2016a, 2017], Schrittwieser et al. [2019], Hessel et al. [2021]), and a general algorithm for *optimistic policy gradients* via *meta-gradient optimization* (common across the algorithmic implementations of Zahavy et al. [2020], Flennerhag et al. [2021]). Using acceleration for (functional) policy gradients is under-explored. This unifying template can be used to design other accelerated policy optimization algorithms, or guide the investigation into other collective properties of these methods.

Leveraging theoretical insights from Sec. 3, in Sec. 3.2, we introduce an optimistic policy gradient algorithm that is adaptive via meta-gradient learning. In Sec. 3.2.1, we use an illustrative task to test several theoretical predictions empirically. First, we tease apart the role of optimism in forward search algorithms. Second, we analyze the properties of the optimistic algorithm we introduced in Sec. 3.2.

2 Preliminaries & notation

Notation Throughout the manuscript, we use $\doteq$ to distinguish a definition from standard equivalence, the shorthand notation $\nabla_x f(x_t) \doteq \nabla_x f(x)|_{x=x_t}$, $\langle \cdot, \cdot \rangle$ denotes a dot product between the arguments. The notation $\lceil \cdot \rfloor$ indicates that gradients are not backpropagated through the targets.

2.1 Markov Decision Processes

We consider a standard reinforcement learning (RL) setting described by means of a discrete-time infinite-horizon discounted Markov decision process (MDP) [Puterman, 1994] $\mathcal{M} \doteq \{\mathcal{S}, \mathcal{A}, r, P, \gamma, \rho\}$, with state space $\mathcal{S}$ and action space $\mathcal{A}$, discount factor $\gamma \in [0, 1)$, with initial states sampled under the initial distribution ρ, assumed to be exploratory $\rho(s) > 0, \forall s \in \mathcal{S}$.

The agent follows an online learning protocol: at timestep $t \geq 0$, the agent is in state $S_t \in \mathcal{S}$, takes action $A_t \in \mathcal{A}$, given a policy $\pi_t(\cdot|s_t)$—the distribution over actions for each state $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$, with $\Delta_{\mathcal{A}}$—the action simplex—the space of probability distributions defined on $\mathcal{A}$. It then receives a reward $R_t \sim r(S_t, A_t)$, sampled from the reward function $r : \mathcal{S} \times \mathcal{A} \to [0, R_{\max}]$, and transitions to a next state $S_{t+1} \sim P(\cdot|S_t, A_t)$, sampled under the transition probabilities or dynamics P. Let $d_\pi(s)$ be a measure over states, representing the discounted visitation distribution (or discounted fraction of time the system spends in a state s) $d_\pi(s) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \Pr(S_t = s | S_0 \sim \rho, A_k \sim \pi(\cdot|S_k), \forall k \leq t)$, with $\Pr(S_t = s|S_0 \ldots)$ the probability of transitioning to a state at timestep t given policy π.

The RL problem consists in finding a policy π maximizing the discounted return

$$J(\pi) \doteq \mathbb{E}_{S \sim \rho}[V_\pi(S)] = (1-\gamma)\mathbb{E}_{\pi,\rho}\left[\sum_{t\geq 0} \gamma^t R_{t+1}\right] \quad \textit{(the policy performance objective)} \quad (1)$$

where $V_\pi \in \mathbb{R}^{|\mathcal{S}|}$ is the value function, and $Q_\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$ the action-value function of a policy $\pi \in \Pi = \{\pi \in \mathbb{R}_+^{|\mathcal{S}| \times |\mathcal{A}|} | \sum_{a \in \mathcal{A}} \pi(s, a) = 1, \forall s \in \mathcal{S}\}$, s.t. (such that) $Q_\pi(s, a) \doteq \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t R_t | S_0 = s, A_0 = a\right]$, and $V_\pi(s) \doteq \mathbb{E}_\pi[Q(s, A)]$. Let $\mathcal{T}_\pi : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ be the Bellman evaluation operator, and $\mathcal{T} : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ the Bellman optimality operator, s.t. $(\mathcal{T}_\pi V)(s) \doteq r(s, \pi(s)) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, \pi(s))V(s')$, and $(\mathcal{T}V)(s) \doteq \max_{a \in \mathcal{A}} r(s, a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, a)V(s') = \max_{\pi \in \Pi}(\mathcal{T}_\pi V)(s)$, with Q-function (abbr. Q-fn) analogs.

2.2 Policy Optimization Algorithms

The classic **policy iteration (PI)** algorithm repeats consecutive stages of (i) one-step greedy policy improvement w.r.t. a value function estimate $\pi_{t+1} \in \mathcal{G}(V_{\pi_t}) \doteq \{\pi : \mathcal{T}_\pi V_{\pi_t} = \mathcal{T} V_{\pi_t}\}$, with $\mathcal{G}$ the greedy set of V_{π_t}, followed by (ii) evaluation of the value function w.r.t. the greedy policy $V_{\pi_{t+1}} = \lim_{m \to \infty} \mathcal{T}_{\pi_{t+1}}^m V_{\pi_t}$. Approximations of either steps lead to **approximate PI (API)** [Scherrer et al., 2015]. Relaxing the greedification leads to **soft PI** [Kakade and Langford, 2002] $\pi_{t+1} \doteq (1-\alpha)\pi_t + \alpha\pi_{t+1}^+$, with $\pi_{t+1}^+ \doteq \arg\max_{\pi \in \Pi} \langle Q_{\pi_t}, \pi \rangle$, for $\alpha \in (0, 1]$, a (possibly time-dependent) step size. **Optimistic PI (OPI)** [Bertsekas and Tsitsiklis, 1996] relaxes the evaluation step instead to $Q_{t+1} \doteq Q_t - \lambda[Q_t - \mathcal{T}Q_t]$. Others [Smirnova and Dohmatob, 2020, Asadi et al., 2021] have extended these results to deep RL and or alleviated assumptions.

More commonly used in practice are **policy gradient** algorithms. These methods search over policies using surrogate objectives $\ell_t(\pi)$ that are local linearizations of the performance $\ell_t(\pi) \doteq J(\pi_t) + \langle \pi, \nabla_\pi J(\pi_t) \rangle - {}^1\!/_{2c}\|\pi - \pi_t\|_\Omega^2$, rely on the true gradient ascent direction of the previous policy in the sequence $\nabla_\pi J(\pi_t)$, and lower bound the policy performance [Agarwal et al., 2019, Li et al., 2021, Vaswani et al., 2021] when $J(\pi)$ is $\frac{1}{\alpha}\Omega$-relatively convex w.r.t. the policy π. As $\alpha \to \infty$ (the regularization term tends to zero), $\pi_{t+1} = \arg\max_{\pi \in \Pi} \ell_t(\pi)$ converges to the solution of ℓ_t, which is exactly the policy iteration update. For intermediate values of α, the **projected gradient ascent** update decouples across states and takes the following form for a direct policy parametrization: $\pi_{t+1} \doteq \mathcal{P}_\Pi(\pi_t + \alpha Q_{\pi_t})$.

Generally, the methods employed in practice extend the policy search to parameterized policy classes with softmax transforms $\Pi_\Theta \doteq \left\{\pi_\theta \middle| \pi_\theta(s, a) = {}^{\exp z_\theta(s,a)}\!/_{\sum_a \exp z_\theta(s,a)} \forall s \in \mathcal{S}, a \in \mathcal{A}, \theta \in \Theta \subset\right.$

$\mathbb{R}^m\}$, with $z_\theta \doteq f(\theta)$, and f a differentiable function, either tabular $z_\theta(s,a) \doteq \theta_{s,a}$, log-linear $z_\theta(s,a) \doteq \phi(s,a)^\top\theta$, with ϕ a feature representation, or neural parametrizations (z_θ-a neural network) [Agarwal et al., 2019]. These methods search over the parameter vector θ of a policy $\pi_\theta \in \Pi_\Theta$. *Actor-critic* methods approximate the gradient direction with a parametrized critic $Q_{w_t} \approx Q_{\pi_t}$, with parameters $w \in \mathcal{W} \subset \mathbb{R}^m$, yielding $\theta_{t+1} \doteq \arg\max_{\theta\in\Theta} \ell(\pi_\theta, Q_{w_t})$, with the surrogate objective $\ell(\pi_\theta, Q_{w_t}) \doteq \langle \pi_\theta, d_{\pi_{\theta_t}}^\top Q_{w_t}\rangle - 1/\alpha\, \mathrm{KL}_{[d_{\pi_{\theta_t}}]}(\pi_\theta, \pi_{\theta_t})$, where we denoted $\mathrm{KL}_{[d]}(\pi,\mu) \doteq \sum_s d(s)\sum_a \pi(a|s)(\log\pi(a|s) - \log\mu(a|s))$ the weighted KL-divergence. The projected gradient ascent version of this update just uses the projection associated with the softmax transform $\pi_{t+1} \doteq \mathrm{KL}(\pi, {}^{\exp z_{t+1/2}}/_{\sum_{\mathcal{A}}\exp z_{t+1/2}})$ with $z_{t+1/2} = z_t + \alpha Q_{\pi_t}$ a target-based update.

Acceleration When the effective horizon γ is large, close to 1 the number of iteration before convergence of policy or value iteration, scales on the order $\mathcal{O}\left({}^1/_{1-\gamma}\right)$. Each iteration is expensive in the number of samples. One direction to accelerate is designing algorithms convergent in a smaller number of iterations, resulting in significant empirical speedups. **Anderson acceleration** Anderson [1965] is an iterative algorithm that combines information from previous iterations to update the current guess, and allows speeding up the computation of fixed points. Anderson Acceleration has been described for value iteration in Geist and Scherrer [2018], extensions to Momentum Value Iteration and Nesterov's Accelerated gradient in Goyal and Grand-Clement [2021], and to action-value (Q) functions in Vieillard et al. [2019]. In the following, we present a policy optimization algorithm with a similar interpretation.

Model-based policy optimization (MBPO) MBPO algorithms based on forward search rely on approximate online versions of multi-step greedy improvement implemented via Monte Carlo Tree Search (MCTS) [Browne et al., 2012]. These algorithms replace the one-step greedy policy in the improvement stage of PI with a multi-step greedy policy Cf. Grill et al. [2020], relaxing the hard greedification, and adding approximations over parametric policy classes, forward search algorithms at scale, can be written as the solution to a regularized optimal control problem, by replacing the gradient estimate in the regularized policy improvement objectives $\ell(\pi)$ of actor-critic algorithms with Q-values Q_{η_t} resulting from approximate lookahead search $\theta_{t+1} \doteq \arg\max_{\theta\in\Theta}\langle\pi, Q_{\eta_t}\rangle_{d_{\pi_t}} - 1/\alpha\,\mathrm{KL}_{[d_{\pi_t}]}(\pi, \pi_t)$, where $Q_{\eta_t} \doteq r^n_{\eta_t}(s,a) + \gamma\sum_{s'} P^n_{\eta_t}(s'|s,a)\sum_{a'}\pi(a'|s')Q_{w_t}(s',a')$ results from using an approximate model-based multi-step operator, obtained via simulators, $\eta = \varnothing$ [Silver et al., 2016a, 2017], or models, with $\eta \in \mathbb{R}^m$ the target parameters [Schrittwieser et al., 2019],

Meta-gradient policy optimization (MGPO) In MGPO [Xu et al., 2018, Zahavy et al., 2020, Flennerhag et al., 2021] the policy improvement step uses a parametrized recursive algorithm $\pi_{\theta_{t+1}} = \varphi(\eta_t, \pi_{\theta_t})$ with $\eta \in \mathbb{R}^n$ the algorithm's (meta-)parameters. For computational tractability, we generally apply inductive biases to limit the functional class of algorithms the meta-learner searches over, e.g., to gradient ascent (GA) parameter updates $\theta_{t+1} = \theta_t + g_{\eta_t}$. The meta-parameters η can represent, e.g., inializations [Finn et al., 2017], losses [Sung et al., 2017, Wang et al., 2019, Kirsch et al., 2019, Houthooft et al., 2018, Chebotar et al., 2019, Xu et al., 2020], internal dynamics [Duan et al., 2016], exploration strategies [Gupta et al., 2018, Flennerhag et al., 2021], hyperparameters [Veeriah et al., 2019, Xu et al., 2018, Zahavy et al., 2020], and intrinsic rewards [Zheng et al., 2018]. The meta-learner's objective is to adapt the parametrized optimization algorithm based on the learner's post-update performance $J(\pi_{\theta_{t+1}})$—unknown in RL, and replaced with a surrogate objective $\ell(\pi_{\theta_{t+1}})$. Zahavy et al. [2020] uses a linear model, whereas Flennerhag et al. [2021] a quasi-Newton method [Nocedal and Wright, 2006, Martens, 2014] by means of a trust region with a hard cut-off after h parameter updates.

3 Acceleration in Policy Optimization

We introduce a simple template for accelerated policy optimization algorithms, and analyze its properties for finite state and action MDPs, tabular parametrization, direct and softmax policy classes. Thereafter, we describe an algorithmic variation suitable for practical and scalable policy gradient algorithms, adaptive via meta-gradient learning.

3.1 A general template for accelerated policy optimization

Consider finite state and action MDPs, and a tabular policy parametrization. The following policy classes will cause policy gradient updates to decouple across states since $\Pi \equiv \Delta_{\mathcal{A}} \times \ldots \Delta_{\mathcal{A}}$ —the $|\mathcal{S}|$-fold product of the probability simplex: (i) the *direct policy representation* using a policy class consisting of all stochastic policies $\pi \in \Pi = \{\pi \in \mathbb{R}_+^{|\mathcal{S}|\times|\mathcal{A}|} \,|\, \sum_{a\in\mathcal{A}}\pi(s,a) = 1, \forall s \in \mathcal{S}\}$, and (ii) the *softmax policy representation* $\pi \in \Pi_\Theta \doteq \left\{\pi \middle| \pi(s,a) = {}^{\exp z(s,a)}/_{\sum_a \exp z(s,a)}, \forall s \in \mathcal{S}, a \in \mathcal{A}\right\}$, with z a dual target, the logits of a policy before normalizing them to probability distributions. We denote the logarithm function, the inverse for the exponential function used by the softmax transform with $z = (\nabla\Omega)^{-1}(\pi)$, and conversely we have $\pi = \nabla\Omega(z)$. For the direct parametrization, we have $z = \pi$, and $\nabla\Omega(z)$ the identity mapping.

Let $\{\pi_t\}_{t\geq 0}$, be a policy sequence, and $\{z_t\}_{t\geq 0}$, $z_t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}, \forall t \geq 0$, the unconstrained targets $z = (\nabla\Omega)^{-1}(\pi)$. A new policy is obtained by projecting $\nabla\Omega(z)$ onto the constraint set induced by the probability simplex, using a projection operator $\pi \doteq \mathcal{P}_\Pi \nabla\Omega(z)$. Let $\{g_t\}_{t\geq 0}$ be (functional) policy gradients, $g \doteq \nabla_\pi J(\pi)$, and $\hat{g}_t \approx g_t$ approximations. Let $\{u_t\}_{t\geq 0}$ be a sequence of (functional) policy updates, described momentarily.

Base algorithm Iterative methods decompose the original multi-iteration objective in Eq. 1 into single-iteration surrogate objectives $\{\ell_t(\pi)\}_{t\geq 0}$, which correspond to finding a maximal policy improvement policy π_{t+1} for a single iteration $\pi_{t+1} = \arg\max_{\pi \in \Pi} \ell_t(\pi)$ and following π_t thereafter. We consider first-order surrogate objectives

$$\pi_{t+1} \doteq \arg\max_{\pi\in\Pi} \ell_t(\pi, u_t) \qquad \ell_t(\pi, u) \doteq \langle \pi, u\rangle - {}^1/_\alpha \|\pi - \pi_t\|^2_\Omega \qquad \textit{(local surrogate objective)} \quad (2)$$

with α a step size, and $\|\cdot\|_\Omega$ the policy distance measured in the norm induced by the policy transform $\nabla\Omega$ (L_1 for the direct policy parametrization on the simplex, and KL-divergence $\mathrm{KL}(\cdot,\cdot)$ for the softmax, cf. Agarwal et al. [2019]). At optimality, we obtain projected gradient ascent in the dual target space

$$\pi_{t+1} \doteq \mathcal{P}_{\Delta^{|\mathcal{A}|}}\left(\nabla\Omega(z_{t+1/2})\right) \qquad z_{t+1/2} = z_t + \alpha u_t \qquad \textit{(policy improvement)} \quad (3)$$

with $z_t \doteq (\nabla\Omega)^{-1}(\pi_t)$, and $\mathcal{P}_\Pi$ the projection operator for the associated norm: L_1 norm for the direct policy parametrization, or the KL for the softmax policy transform, cf. Agarwal et al. [2019]. It is known that for the softmax parametrization the closed form update results in the natural policy gradient/mirror ascent/proximal update, and can be written in closed form as $\pi_{t+1} \propto \pi_t \exp(\alpha u_t)$.

Acceleration If the update rule u_t returns just an estimation of the standard gradient $\hat{g}_t \approx g_t$, s.t. $u_t \doteq \hat{g}_t$, then the algorithm reduces to the inexact NPG/mirror ascent/proximal update $\pi_{t+1} \doteq \mathcal{P}_{\Delta^{|\mathcal{A}|}}(\nabla\Omega)^{-1}(z_t + \alpha\hat{g}_t)$. The inductive bias is fixed and does not change with past experience, and so acceleration is not possible. If the update rule is auto-regressive, the inductive bias formed is similar to the canonical **momentum** algorithm—Polyak's Heavy-Ball method [Polyak, 1964],

Algorithm 1 *Accelerated policy gradients*

```
for $t = 1, 2 \ldots T$ do
    ▷policy evaluation: estimate $\hat{g}_t \approx g_t$
    ▷acceleration: update $u_t$ with momentum Eq. 4, optimism+lookahead Eq. 5, or extra-gradients Eq. 6
    ▷policy improvement: update $\pi_{t+1}$ with Eq. 3
end for
```

$$u_t \doteq \mu u_{t-1} + \beta\hat{g}_t \implies z_{t+1/2} = z_t + \mu(z_t - z_{t-1/2}) + \alpha\beta\hat{g}_t \qquad \textit{(momentum/Polyak's Heavy-Ball)} \quad (4)$$

with β and μ step-size, and momentum decay. Because Heavy Ball carries momentum from past updates, it can encode a model of the learning dynamics that leads to faster convergence.

Optimism A typical form of optimism is to predict the next gradient in the sequence $\hat{g}_{t+1} \approx g_{t+1}$, while simultaneously subtracting the previous prediction $\hat{g}_t$, thereby computing at each iteration $u_t = \beta\hat{g}_{t+1} + \mu[u_{t-1} - \beta\hat{g}_t]$, and updating $\pi_{t+1} = \mathcal{P}_\Pi(\nabla\Omega)^{-1}(z_t + \alpha u_t)$. The policy updates are **extrapolations** based on predictions of next surrogate objective in the sequence

$$u_t \doteq \beta\hat{g}_{t+1} + \mu[u_{t-1} - \beta\hat{g}_t] \implies z_{t+1/2} = z_t + \mu(z_t - z_{t-1/2}) + \alpha\beta\hat{g}_{t+1} - \alpha\beta\hat{g}_t \qquad \textit{(optimism)} \quad (5)$$

If the gradient predictions $\{\hat{g}_{t+1}\}_{t\geq 0}$ are accurate $\hat{g}_{t+1} = g_{t+1}$, the optimistic update rule can accelerate. Using $u_{t-1} = g_t$ we obtain the predictor-corrector approach. But in RL, agents generally do not have g_t, so the distance to the true gradient $\|g_t - u_{t-1}\|_*$ will depend on how good the prediction $\hat{g}_t$ was at the previous iteration $\|g_t - \hat{g}_t\|_*$, with $\|\cdot\|_*$ the dual norm. Since we have not computed π_{t+1} at time t, and we do not have the prediction $\hat{g}_{t+1}$, existing methods perform the following techniques.

Lookahead *Model-based* policy optimization methods look ahead, one-step or multiple steps ($n \geq 1$), using a model or simulator ($\hat{r}$, $\hat{P}$), to compute $\hat{g}_{t+1} - \hat{g}_t$. Using an empirical distribution over states and actions this results in regressing Q-fn residuals $Q_{t+1} - Q_t$, and setting $Q_{t+1} \doteq \mathcal{T}Q_t$, means using the Q-learning residuals $U_t = \mu U_{t-1} + \beta[\hat{r}^n_t(s,a) + \gamma\sum_{s'}\hat{P}^n_t(s'|s,a)\sum_{a'}\max_{a'} Q_t(s',a') - \mu Q_t(s',a')$.

Extra-gradients *Optimistic meta-learning* algorithms are essentially extra-gradient algorithms, since they use the previous prediction u_{t-1} (or momentum) as a proxy to compute a *half-step proposal* $\pi_{t+1/2} = \mathcal{P}_\Pi(\nabla\Omega)^{-1}(z_t + \alpha u_{t-1})$, compute $\hat{g}_{t+1/2} \doteq \nabla_\pi J(\pi_{t+1/2})$ (with a model-based procedure for sample efficiency) and retrospectively obtain the optimistic update and a new target policy $\pi_{t+1} = \mathcal{P}_\Pi(\nabla\Omega)^{-1}(z_t + \alpha u_t)$

$$u_t \doteq \beta\hat{g}_{t+1/2} + \mu[u_{t-1} - \beta\hat{g}_t] \implies z_{t+1/2} = z_t + \mu(z_t - z_{t-1/2}) + \alpha\beta\hat{g}_{t+1/2} \quad \textit{(extra-gradients)} \tag{6}$$

Practical implementations however do not compute the target policy retrospectively, but rather use the half-step proposal at the next iteration. Alg. 1 summarizes the procedure.

3.2 Towards a practical Accelerated Policy Gradient algorithm

More commonly used in practice are neural, or log-linear parametrizations for actors, and equivalent parametrizations of gradient-critics (e.g., Espeholt et al. [2018], Schulman et al. [2015, 2017], Abdolmaleki et al. [2018], Tomar et al. [2020], Zahavy et al. [2020], Flennerhag et al. [2021], Hessel et al. [2021]).

Consider a parameterized softmax policy class $\Pi_\Theta \doteq \left\{\pi_\theta | \pi_\theta(s,a) \doteq {}^{\exp z_\theta(s,a)}/_{\sum_a \exp z_\theta(s,a)}, \forall s \in \mathcal{S}, a \in \mathcal{A}, \theta \in \Theta \subset \mathbb{R}^m\right\}$, with $\pi_\theta \in \Pi_\Theta$, and z_θ a differentiable logit function. Let u_η represent a parametric class of policy updates, with parameters $\eta \in \mathbb{R}^{m'}$, which we discuss momentarily.

Base algorithm We recast the policy search in Eq. 2 over policy parameters θ

$$\theta_{t+1} \doteq \arg\max_{\theta\in\Theta} \ell_t(\pi_\theta, u_{\eta_t}) \qquad \ell_t(\pi_\theta, u_\eta) \doteq \langle \pi_\theta, u_\eta \rangle - {}^1/_\alpha \, \mathrm{KL}_{[d_{\pi_{\theta_t}}]}(\pi_\theta, \pi_{\theta_t}) \tag{7}$$

Using function composition, we write the policy improvement step using a parametrized recursive algorithm $\pi_{\theta_{t+1}} = \varphi(\eta_t, \pi_{\theta_t})$ with η the algorithm's (meta-)parameters. We assume the $\varphi(\eta, \cdot)$ is differentiable and smooth w.r.t. η. If Eq. 7 can be solved in closed form, an alternative is to compute the non-parametric closed-form solution $\pi_{t+1} \propto \pi_{\theta_t} \exp \alpha u_{\eta_t}$ and (approximately) solve the projection $\theta_{t+1} \doteq \arg\min_{\theta\in\Theta} \mathrm{KL}(\pi_\theta, \pi_{t+1})$. For both approaches, we may use $h \geq 1$ gradient steps on Eq. (7)

$$\theta_t^{k+1} = \theta_t^k + \xi y_{t+1} \qquad y_{t+1} \doteq \nabla_\theta \ell_t(\pi_{\theta_t^k}, u_{\eta_t}) \qquad \forall k \in [0..h), \theta_t^0 \doteq \theta_t \qquad \theta_{t+1} \doteq \theta_t^h \tag{8}$$

with ξ a parameter step size. By function compositionality, we have $\nabla_\theta \ell(\pi, g) = \nabla_\theta \pi_\theta^\top \nabla_\pi \ell(\pi, g)$. This part of the gradient $\nabla_\theta \pi_\theta(s,a) = \pi_\theta(s,a)\nabla_\theta \log \pi_\theta(s,a)$ is generally not estimated, available to RL agents, and computed by backpropagation. Depending on how the other component $\nabla_\pi \ell(\pi, g)$ is defined, in particular u_{η_t}, we may obtain different algorithms. Generally, this quantity is expensive to estimate accurately in the number of samples for RL algorithms.

In order to admit an efficient implementation of the parametrized surrogate objective in Eq. 7, we only consider separable surrogate parametrizations over the state space. We resort to sampling experience under the empirical distribution $\hat{d}$ and the previous policy π_{θ_t}, rather than using the stationary distribution, and we replace the expectation over states and actions in the gradient with an empirical average over rollouts or mini-batches $\mu \doteq \{(S^0, A^0), (S^1, A^1)\dots\}$. Leveraging this compositional structure, the algorithms we consider use weighted policy updates $u_\eta(s,a) = \hat{d}(s)U_\eta(s,a)$

Algorithm 2 Accelerated Policy Gradients
——————[a practical algorithm]——————

```
input: $(\theta_0, \eta_0)$, predictions $\{Q_t\}_{t\geq 0}$
for each iteration $t = 1, 2\dots$ do
    Sample from $\pi_{\theta_t}$ & store in buffer $\mu$
    # policy improvement
    Update $\theta_{t+1}$ with Eq. 9
    # acceleration
    Compute $\pi_{t+2}$ with $Q_{t+1}$ in Eq. 7 & $\mu$
    Update $\eta_{t+1}$ using with Eq. 11 & $\mu$
end for
```

$$\ell_t(\pi_\theta, U_\eta) \doteq \mathbb{E}_\mu[{}^{\pi_\theta(A|S)}/_{\pi_{\theta_t}(A|S)} U_\eta(S,A) - {}^1/_\alpha \, \mathrm{KL}(\pi_\theta(\cdot|S), \pi_{\theta_t}|(\cdot|S)] \tag{9}$$

Example 1 **(A non-optimistic algorithm)** Under this definition, the standard actor-critic algorithm uses $U_\eta \doteq Q_\eta$ and updates η with semi-gradient temporal-difference (TD) algorithms, by taking $h \geq 1$ steps of gradient descent on the following proxy objective for the value error $\eta_{t+1} \doteq \arg\min_{\eta\in\mathbb{R}^{m'}} f_t(\eta, Q_{t+1})$, where

$$f_t(\eta, Q_{t+1}) \doteq \mathbb{E}_\mu[{}^1/_2\big(\lceil Q_{t+1} \rfloor(S,A) - Q_\eta(S,A)\big)^2] + {}^1/_{2\zeta}\|\eta - \eta_t\|_2^2 \tag{10}$$

with ζ a step size. The targets Q_{t+1} are typically bootstrapped from Q_{η_t}.

Acceleration We replace the standard policy update with an optimistic decision-aware update

$$f_t(\eta, Q_{t+1}) \doteq \beta\ell_{t+1}(\pi_{t+2}, Q_{t+1}) + \mu[\ell_t(\varphi(\eta, \pi_{\theta_t}), U_{\eta_{t-1}}) - \beta\ell_t(\pi_{\theta_{t+1}}, Q_t)]$$

where we used the notation $\pi_{\theta_{t+1}} \doteq \varphi(\eta_t, \pi_{\theta_t})$ which denotes the policy improvement step uses a parametrized recursive algorithm with parameters η as indicated in the update rule U_η. $\varphi(\eta, \pi_{\theta_t})$. The algorithm we use computes $\pi_{\theta_{t+1}} = \mathcal{P}_{\Pi_\Theta}\pi_{t+1}$, from optimal policies $\pi_{t+1} \doteq \arg\max_\pi \ell_t(\pi, U_{\eta_{t-1}}) \propto \pi_{\theta_t}\exp(\alpha U_{\eta_{t-1}})$, and $\pi_{t+2} \doteq \arg\max_\pi \ell_{t+1}(\pi, Q_{t+1}) \propto \pi_{\theta_{t+1}}\exp(\alpha Q_{t+1})$ cf. Eq.8.

With $\ell_t(\varphi(\eta, \pi_{\theta_t}), U_{\eta_{t-1}}) = \mathrm{KL}(\varphi(\eta, \pi_{\theta_t}), \pi_{\theta_{t+1}})$, $\ell_{t+1}(\pi_{t+2}, Q_{t+1}) - \ell_t(\pi_{\theta_{t+1}}, \beta Q_t) = \mathrm{KL}(\pi_{\theta_{t+1}}, \pi_{t+2})$, and $\ell_t(\varphi(\eta, \pi_{\theta_t}), \beta Q_{t+1} + [U_\eta - \beta Q_t]) = \mathrm{KL}(\pi_{\theta_t}, \pi_{t+2})$, the left-hand side of the generalized Pythagorean theorem restates the objective above for $\varphi(\eta, \pi_{\theta_t}) = \varphi(\eta_t, \pi_{\theta_t})$

$$\mathrm{KL}(\pi_{\theta_t}, \pi_{\theta_{t+1}}) + \mathrm{KL}(\pi_{\theta_{t+1}}, \pi_{t+2}) = \mathrm{KL}(\pi_{\theta_t}, \pi_{t+2}) + \langle \nabla_\pi \mathrm{KL}(\pi, \pi_{t+2})|_{\pi=\pi_{\theta_{t+1}}}, \pi_{\theta_t} - \pi_{\theta_{t+1}}\rangle$$

By cosine law, if $\langle \nabla_\pi \mathrm{KL}(\pi, \pi_{t+2})|_{\pi=\pi_{\theta_{t+1}}}, \pi_{\theta_t} - \varphi(\eta, \pi_{\theta_t})\rangle \geq 0$ we can move η, and indirectly, $\varphi(\eta, \pi_{\theta_t})$ in the opposite direction of π_{θ_t} to decrease $\mathrm{KL}(\varphi(\eta, \pi_{\theta_t}), \pi_{t+2})$ retrospectively. We now want to find η which minimizes $\langle \nabla_\pi \mathrm{KL}(\pi_{\theta_{t+1}}, \pi_{t+2}), \varphi(\eta, \pi_{\theta_t})\rangle$. We use a first-order method, to optimize the linearization of this objective

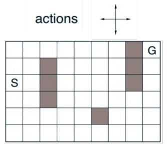
$$f_t(\eta, Q_{t+1}) \doteq \langle \eta, \nabla_\eta\varphi(\eta_t, \pi_{\theta_t})^\top \nabla_\pi \mathrm{KL}(\pi_{\theta_{t+1}}, \pi_{t+2})\rangle + {}^1\!/_{2\zeta}\|\eta - \eta_t\|_2^2 \qquad (11)$$

where π_{t+2} depends on Q_{t+1}. Alg. 2 summarizes the procedure.

Next, we empirically study: (i) the effect of grounded meta-optimization targets Q_{t+1} based on true optimistic predictions $Q_{t+1} \doteq Q_{\pi_{\theta_{t+1}}}$, and (ii) using self-supervised, inaccurate predictions—obtained with a separate estimator: $Q_{t+1} \doteq Q_{w_{t+1}}$, with w separately learned parameters. Bootstrapping the meta-optimizer on itself $Q_{t+1} \doteq U_{\eta_t}$ quickly diverges without proper grounding. We leave to future work the exploration of other ways of adding partial feedback to ground the bootstrap targets, e.g. using stochastic truncated bootstrap targets $\hat{\mathcal{T}}^n U_{\eta_t}$.

3.2.1 Illustrative empirical analysis

In this section, we investigate acceleration using optimism for online policy optimization, in an illustrative task. We mentioned one option for computing optimistic predictions $\{Q_t\}_{t\geq 0}$ is using a model or simulator. Consequently, in Sec. 3.2.2, we begin with a brief study on the effect of the lookahead horizon on the optimistic step, in order to understand the acceleration properties of multi-step forward search algorithms, and distinguish between two notions of optimism. Thereafter, in Sec. 3.2.3, we consider the accelerated policy gradient algorithm we designed in Sec. 3.2 (summarized in Alg. 2), and investigate emerging properties for several choices of policy targets π_{t+2} obtained with optimistic predictions $\{Q_{t+1}\}_{t\geq 0}$.



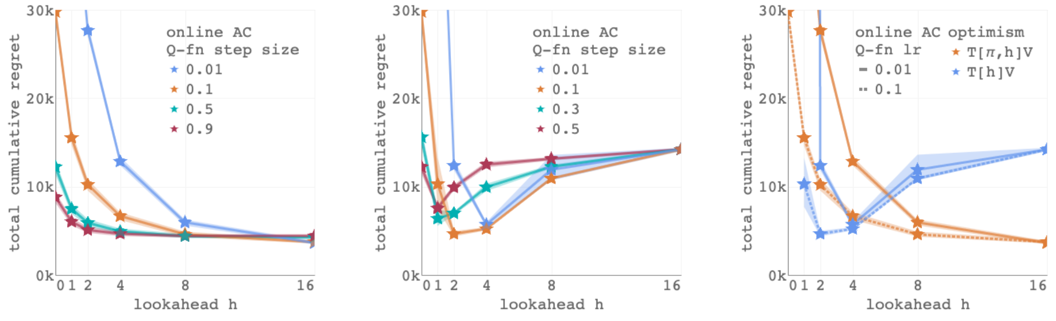
Experimental setup In both experiments, we use the discrete navigation task from [Sutton and Barto, 2018], illustrated aside (details in Appendix C).

3.2.2 Optimism with multi-step forward search

For this illustration, we use exact simulators. The only source of inaccuracy in the gradient prediction is the depth truncation from using a fixed lookahead horizon h. We use this experiment to show the difference between: (i) optimism within the local policy evaluation problem ($Q_{t+1} \doteq \mathcal{T}^h_{\pi_t} Q_t$), and (ii) optimism within the global maximization problem ($Q_{t+1} \doteq \mathcal{T}^h Q_t$).

Algorithms We consider an online AC algorithm, with forward planning up to horizon h for computing the search Q-values $Q_{t+1} \doteq \mathcal{T}^h_{\pi_b} Q_{w_t}$, bootstrapping at the leaves on Q_{w_t}, trained with using Eq. 10, and π_b, a tree-search policy. We optimize the policy π_θ online, using $h = 1$ gradient steps on Eq 9: $\theta_{t+1} = \theta_t + \beta\nabla_\theta \log \pi_{\theta_t}(A|S)\mathrm{A}_{t+1}(S, A)$, with actions sampled online and $\mathrm{A}_{t+1} \doteq Q_{t+1} - V_{t+1}$ the h-step advantage, where $V_{t+1}(S) \doteq \mathbb{E}_{\pi_{\theta_t}}(\cdot|S)[Q_{t+1}(S, A)]$.

Relationship between acceleration and optimistic lookahead horizon We use a multi-step operator in the optimistic step, which executes a Q-fn lookahead expansion up to horizon h, and tree back-up. For the tree policy π_b, we distinguish between: (i) extra policy evaluation steps with the previous policy, $Q_{t+1} \doteq \mathcal{T}^h_{\pi_t} Q_{w_t}$ (Fig 2(a)), and (ii) extra greedification steps, $Q_{t+1} \doteq \mathcal{T}^h Q_{w_t}$ (Fig 2(b)). The policy & Q-fn learners are trained online with $\theta_{t+1} = \theta_t + \xi y_t$, s.t. $\mathbb{E}_{\hat{d},\pi_t}[y_t] \doteq \mathbb{E}_{\hat{d},\pi_t}[\mathrm{A}_{t+1}(S, A)\nabla_\theta \log \pi_{\theta_t}(A|S))]$, with $\mathrm{A}_{t+1} = Q_{t+1} - V_{t+1}$, where $V_{t+1}(S) \doteq$



(a) Optimistic evaluation (b) Optimistic improvement (c) Difference between (a) and (b)

Figure 1: Optimism with extra steps of forward search with a simulator. x-axis: lookahead horizon h; y-axis: total cumulative regret $\sum_{k \le t} J(\pi^*) - J(\pi_k)$. Lookahead targets $r + \gamma Q_{t+1}$ are used for: **(a) Optimistic evaluation** $Q_{t+1} \doteq \mathcal{T}^h_{\pi_t} Q_{w_t}$. Increasing the optimistic lookahead horizon h helps, and a horizon $h = 0$ is worst. Colored curves denote the step size ζ used to learn the parameter vector w of Q_w with online TD(0). The step size controls the quality of the gradient via the accuracy of the search Q-values Q_{t+1} (more details in the main text). Shades denote confidence intervals over 10 runs. **(b) Optimistic improvement** $Q_{t+1} \doteq \mathcal{T}^h Q_{w_t}$. Intermediate values of the optimistic lookahead horizon h trade off accumulating errors for shorter horizons. **(c) Comparison between the two notions of optimism** local—evaluation within the current prediction problem, and global—improvement within the optimization problem, for two step sizes.

$\mathbb{E}_{\pi_t(\cdot|S)}[Q_{t+1}(S, A)]$. The advantage function A_{t+1} uses search Q-values Q_{t+1}, and critic parameters w trained with Eq. 10 from targets based on the search Q-values $r(S, A) + \gamma Q_{t+1}(S, A)$.

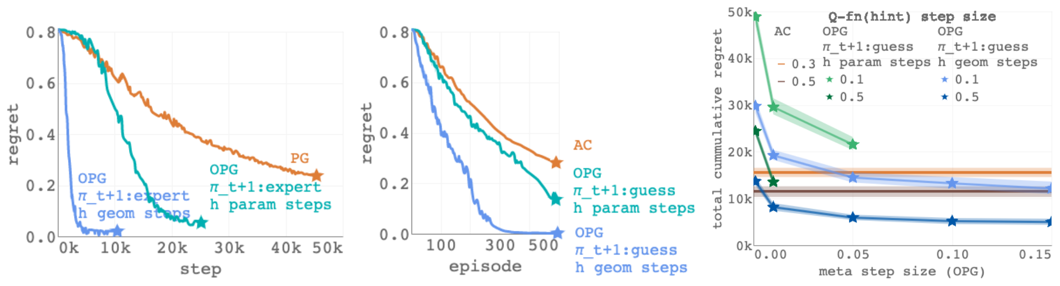
Results & observations Fig. 1(c) shows the difference between optimistic improvement—the gradient prediction has foresight of future policies on the optimization landscape, and optimistic evaluation—the gradient prediction is a refinement of the previous gradient prediction toward the optimal solution to the local policy improvement sub-problem. As Fig. 1(a) depicts, more lookahead steps with optimistic evaluation, can significantly improve inaccurate gradients, where accuracy is quantified by the choice of ζ, the Q-fn step size for w. Thus, for $\pi_b \doteq \pi_t$, increasing $h \to \infty$, takes the optimistic step with the exact (functional) policy gradient of the previous policy, $Q_{t+1} = \mathcal{T}^h_{\pi_b} Q_{w_t} = \mathcal{T}^h_{\pi_t} Q_{w_t} \overset{h\to\infty}{\longrightarrow} Q_{\pi_t}$. As Fig. 1(b) shows, the optimal horizon value for optimistic improvement is another, one that trades off the computational advantage of extra depth of search, if this leads to accumulating errors, as a result of depth truncation, and bootstrapping on inaccurate values at the leaves, further magnified by greedification.

3.2.3 Accelerated policy optimization with optimistic policy gradients

We now empirically analyze some of the properties of the practical meta-gradient based adaptive optimistic policy gradient algorithm we designed in Sec. 3.2 (Alg. 2).

(i) Acceleration with optimistic policy gradients We first remove any confounding factors arising from tracking inaccurate target policies π_{t+2} in Eq.11, and resort to using the true gradients of the post-update performance of $\pi_{\theta_{t+1}}$, $Q_{t+1} \doteq Q_{\pi_{\theta_{t+1}}}$, but distinguish between two kinds of lookahead steps: (a) *parametric*, or (b) *geometric*. This difference is indicative of the farsightedness of the optimistic prediction. In particular, this distinction is based on the policy class of the target, whether it be a (a) parametric policy target $\pi_{\theta_{t+2}}$, obtained using h steps on Eq. 8, with $y_{t+1} \doteq \nabla_\theta \ell_t(\pi_{\theta^k_{t+1}}, Q_{t+1}) \forall k \ge h$, or a (b) non-parametric policy target, $\pi_{t+2} \propto \pi_{\theta_{t+1}} \exp \alpha Q_{t+1}$. The results shown are for $h = 1$, and $\alpha = 1$.

Results & observations When the meta-optimization uses an adaptive optimizer (Adam [Kingma and Ba, 2015]), Fig 2(a) shows there is acceleration when using targets π_{t+2} one step ahead of the learner parametric, or geometric. The large gap in performance between the two optimistic updates owes to the fact that target policies that are one geometric step ahead correspond to steepest directions of ascent, and consequently, may be further ahead of the policy learner in the space of parameters, leading to acceleration. Additional results illustrating sensitivity curves to hyperaparameters are added in Appendix C. When the meta-optimization uses SGD, the performance of the meta-learner algorithms is slower, lagging behind the PG baseline, but the ordering over the optimistic variants is maintained (Fig 5(a) in Appendix C), which indicates that the correlation between acceleration



(a) Expert targets (b) Target predictions (c) Hyperparameter sensitivity

Figure 2: Accelerated policy optimization with optimistic policy gradients (a) x-axis: number of steps, y-axis: regret $J(\pi^*) - J(\pi_t)$. Different colored curves denote: standard PG, *optimistic policy gradients (OPG)* - with parametric target policies, and non-parametric target policies, trained with meta-gradient learning from optimistic predictions using the true post-update gradients. **(b)** x-axis: number of episodes, y-axis: regret. *Optimistic policy gradients (OPG)* are meta-learned from inaccurate optimistic predictions using Q-fn estimations. **(c)** Hyper-parameter sensitivity curves. x-axis: meta-learning rate for η, y-axis: total cumulative regret $\sum_{k \leq t} J(\pi^*) - J(\pi_k)$. The plot shows *optimistic policy gradients* meta-learned from inaccurate optimistic predictions. Different tones depict different accuracies of the optimistic prediction, indirectly quantified via the optimistic Q-fn's step size. Straight lines show a baseline standard AC. Shades denote confidence intervals over 10 runs.

and how far ahead the targets are on the optimization landscape is independent of the choice of meta-optimizer.

(ii) How target accuracy impacts acceleration Next, we relax the setup from the previous experiment, and use inaccurate predictions $Q_{t+1} \approx Q_{\pi_{\theta_{t+1}}}$, instead of the true post-update gradients. In particular, we resort to online sampling under the empirical on-policy distribution $\hat{d}$, and use a standard Q-fn estimats to track the action-value of the most recent policy $Q_{w_{t+1}} \approx Q_{\pi_{\theta_{t+1}}}$ using Eq. 10, with TD(0): $w_{t+1} = w_t - \zeta[r(S, A) + \gamma Q_{w_t}(S, A) - Q_{w_t}(S, A)]\nabla_w Q_{w_t}(S, A)$, with step size ζ. With respect to the policy class of the targets, we experiment with the same two choices (a) *parametric* π_{θ_t+2}, or (b) *non-parametric* π_{t+2} . Targets are ahead of the optimistic learner, in (a) *parameter* steps, for the former, and geometric steps for the latter.

Results & observations Even when the target predictions are inaccurate, Fig.2(b) shows that optimistic policy ascent directions distilled from lookahead targets that use these predictions can still be useful (meta-optimization uses Adam, although promising results are in Appendix C also for meta-optimization with SGD). Non-parametric targets, ahead in the optimization, show similar potential as when using true optimistic predictions. Fig 2(c) illustrates the total cumulative regret (y-axis) stays consistent across different levels of accuracy of the optimistic predictions used by the targets, which is quantified via the Q-fn step sizes (ζ), and indicated by different tones for each algorithm. As expected, we observe parametric targets to be less robust to step size choices, compared to non-parametric ones, analogous the distinct effect of non-covariant gradients vs natural gradients.

4 Concluding remarks

We presented a simple, principled template for accelerating policy optimization algorithms, and connected seemingly distinct classes of algorithms: model-based policy optimization algorithms, and optimistic meta-learning. We drew connections to well-known universal algorithms from convex optimization, and investigated some of the properties of acceleration in policy optimization. We used this interpretation to design an optimistic PG algorithm based on meta-gradient learning, highlighting its features empirically.

Related work We defer an extensive discussion on related work to the appendix. The closest in spirit to this operator-view formulation is the predictor-corrector paradigm, used also by Cheng et al. [2018]. The most similar optimistic algorithm for policy optimization is AAPI [Hao et al., 2020]. Both analyze optimism from a smooth optimization perspective, whereas we focus the analysis on Bellman-operators and PI-like algorithms, optimistic update rules, thus allowing the unification. We extend the empirical analysis of Flennerhag et al. [2021], who only focused on meta-learning hyperparameters of the policy gradient, and used optimistic update rule in parameter space, which is less principled and lacks guarantees. Other meta-gradient algorithms [Sung et al., 2017, Wang et al., 2019, Chebotar et al., 2019, Xu et al., 2020] take to more empirical investigations. We focused on understanding the core principles common across methods, valuable in designing new algorithms in this space, optimistic in spirit.

Future work We left many questions unanswered, theoretical properties, and conditions on guaranteed accelerated convergence. The scope of our experiments stops before function approximation, or bootstrapping the meta-optimization on itself. Conceptually, the idea of optimizing for future performance has applicability in lifelong learning, and adaptivity in non-stationary environments [Flennerhag et al., 2021, Luketina et al., 2022, Chandak et al., 2020].

Acknowledgements

The first author gratefully acknowledges support from FRQNT—Fonds de recherche du Québec, Nature et technologies, and IVADO.

Appendix

A Convergence rates for policy gradient algorithms

Alg	Opt	$\mathbf{\Pi}/\mathbf{\Pi}_\Theta$	Q	Notes	$\mathcal{O}(\mathbf{T})$	Reference
PI	✗	Π	-	convex Π	linear	Ye [2011]
API	✗	Π	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	convex Π	linear	Scherrer [2016]
SoftPI	✗	Π	-	convex Π	linear	Bhandari and Russo [2021]
GA	✗	Π_Θ	-	log barrier reg.	$\mathcal{O}(1/\sqrt{T})$	Agarwal et al. [2019]
GA	✗	Π_Θ	-	-	$\mathcal{O}(1/T)$	Mei et al. [2020b]
GA	✗	Π_Θ	-	entropy reg	linear	Mei et al. [2020b]
PGA	✗	Π_Θ	-	-	$\mathcal{O}(1/\sqrt{T})$	Bhandari and Russo [2019]
PGA	✓	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	adaptive step size	$\mathcal{O}(1/\sqrt{T})$	Cheng et al. [2018]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	(non/)convex Π	$\mathcal{O}(1/\sqrt{T})$	Agarwal et al. [2019]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$		$\mathcal{O}(1/\sqrt{T})$	Shani et al. [2019]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	entropy reg.	$\mathcal{O}(1/T)$	Shani et al. [2019]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	adaptive step size	linear	Khodadadian et al. [2021]
PGA	✗	Π_Θ	-	adaptive step size	linear	Bhandari and Russo [2019]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	adaptive step size	linear	Xiao [2022]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	entropy reg.	linear	Cen et al. [2022]
PGA	✗	Π_Θ	-	entropy reg.	linear	Bhandari and Russo [2021]
PGA	✗	Π_Θ	$\mathbb{R}^{\lvert S\rvert\times\lvert A\rvert}$	strong reg.	linear	Lan [2022]
PGA	✗	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$		$\mathcal{O}(1/T)$	Agarwal et al. [2019]
PGA	✗	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$	adaptive step size	$\mathcal{O}(1/T^{2/3})$	Abbasi-Yadkori et al. [2019]
PGA	✓	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$	adaptive step size	$\mathcal{O}(1/T^{3/4})$	Hao et al. [2020]
PGA	✓	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$	adaptive step size	$\mathcal{O}(1/\sqrt{T})$	Lazic et al. [2021]
PGA	✗	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$	geom incr. step size	linear	Chen and Theja Maguluri [2022]
PGA	✗	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$	geom incr. step size	linear	Alfano and Rebeschini [2023]
PGA	✗	$\Pi_{\Phi^\top\Theta}$	$\Phi^\top\eta$	geom incr. step size	linear	Yuan et al. [2023]

Table 1: Summary of previous work on rates of convergence for policy gradient algorithms (Columns) **Alg**—algorithm used, **Opt**—whether it uses optimism, $\mathbf{\Pi}/\mathbf{\Pi}_\Theta$—policy class, **Q**—the space of the gradient-critic prediction used (if not using the true gradient-critic/Q-fn of the policy performance objective, $\mathbf{Q}_\pi$, in which case it is marked with $-$), $\mathcal{O}(\mathbf{T})$—iteration complexity (finite sample analysis of convergence), as a function of number of iterations $\mathbf{T}$, **Notes**—other assumptions, limitations, observations. **(Algorithm)** the following abbreviations are used: PI—policy iteration, API—approximate policy iteration, SoftPI—soft policy iteration, GA—gradient ascent, PGA—projected gradient ascent (including (inexact) natural gradient ascent, (inexact) mirror ascent, (inexact) dual-averaging, (inexact) primal-dual views). **(Policy class)** the following abbreviations are used: Π—tabular with direct/natural policy parametrization, Π_Θ—tabular with softmax policy parametrization, $\Pi_{\Phi^\top\Theta}$—log-linear policy parametrization, i.e. the softmax transform is applied on a linear parametrization, $\Phi^\top\Theta$, with Φ—the feature representation, and with corresponding linear gradient approximation over the policy's features $Q = \Phi^\top\eta$, with η—parameter vector η.

B Related work

B.1 Optimism in policy optimization

Problem formulation The RL problem consists in finding a policy π maximizing the discounted return—the policy performance objective: $J(\pi) \equiv \mathbb{E}_{S\sim\rho}[V_\pi(S)] = (1-\gamma)\mathbb{E}_{\pi,\rho}\big[\sum_{t\geq 0}\gamma^t R_t\big]$, where $V_\pi \in \mathbb{R}^{|\mathcal{S}|}$ is the value function, and $Q_\pi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$ the action-value function of a policy $\pi \in \Pi = \{\pi \in \mathbb{R}_+^{|\mathcal{S}|\times|\mathcal{A}|} | \sum_{a\in\mathcal{A}}\pi(s,a) = 1, \forall s \in \mathcal{S}\}$, s.t. $Q_\pi(s,a) \equiv \mathbb{E}_\pi\left[\sum_{t=0}^\infty \gamma^t R_t | S_0 = s, A_0 = a\right]$, and $V_\pi(s) \equiv \mathbb{E}_\pi\left[Q(s,A)\right]$.

B.1.1 Policy iteration

Policy iteration The classic **policy iteration** algorithm repeats consecutive stages of (i) one-step greedy policy improvement w.r.t. a value function estimate

$$\pi_{t+1} \in \mathcal{G}(V_{\pi_t}) = \{\pi : \mathcal{T}_\pi V_{\pi_t} = \mathcal{T}V_{\pi_t}\} \iff \pi_{t+1} = \arg\max_{\pi\in\Pi}\langle\nabla J(\pi_t), \pi\rangle = \langle Q_t, \pi\rangle_{d_{\pi_t}} \tag{12}$$

with $\mathcal{G}$ the greedy set of V_{π_t}, followed by (ii) evaluation of the value function w.r.t. the greedy policy

$$V_{\pi_{t+1}} = \lim_{h\to\infty}\mathcal{T}^h_{\pi_{t+1}}V_{\pi_t} \text{ or } Q_{\pi_{t+1}} = \lim_{h\to\infty}\mathcal{T}^h_{\pi_{t+1}}Q_{\pi_t} \tag{13}$$

Approximate policy iteration Approximations of either steps lead to approximate PI (API) [Scherrer et al., 2015], in which we replace the two steps above with

$$\pi_{t+1} \in \mathcal{G}(V_{\pi_t}) = \{\pi : \mathcal{T}_\pi V_{\pi_t} \geq \mathcal{T}V_{\pi_t} - \epsilon_{t+1}\} \tag{14}$$

with ϵ_{t+1} a greedification and/or value approximation error.

Soft policy iteration Relaxing the greedification leads to **soft policy iteration**, or conservative policy iteration [Kakade and Langford, 2002], called Frank-Wolfe by Bhandari and Russo [2021]. The minimization problem decouples across states to optimize a linear objective over the probability simplex

$$\pi_{t+1} = (1-\alpha)\pi_t + \alpha\pi^+_{t+1} \text{ with } \pi^+_{t+1} = \arg\max_{\pi\in\Pi}\langle Q_{\pi_t}, \pi\rangle_{d_{\pi_t}} \tag{15}$$

for $\alpha \in [0,1]$, a (possibly time-dependent) step size, and $\langle\cdot,\cdot\rangle_d$ a state weighting that places weight $d(s)$ on any state-action pair (s,a).

Optimistic policy iteration (OPI) [Bertsekas and Tsitsiklis, 1996] relaxes the evaluation step instead to

$$Q_{t+1} = (1-\lambda)Q_t + \lambda Q^+_{t+1}, \text{ with } Q^+_{t+1} = \mathcal{T}^h_{\pi_{t+1}}Q_t, \forall h \geq 0 \tag{16}$$

with $\lambda \in [0,1]$. Other partial evaluations are possible, such as a geometric interpolation with $\tilde{\lambda}$ (possibly different than λ) of multi-step partial evaluations: $Q^+_{t+1} \equiv \mathcal{T}^{\tilde{\lambda}}_{\pi_t}Q_t = (1-\tilde{\lambda})\sum_{h=0}^\infty \tilde{\lambda}^h\mathcal{T}^{h+1}_{\pi_{t+1}}$, with $\tilde{\lambda} \in [0,1]$.

B.1.2 Policy gradients

Projected Gradient Descent Starting with some policy $\pi \in \Pi$, an iteration of projected gradient ascent with step size α updates to the solution of the regularized problem

$$\pi_{t+1} = \arg\max_\pi\langle\nabla J(\pi_t), \pi\rangle + \frac{1}{\alpha}\sum_{s\in\mathcal{S}}d_{\pi_t}(s)\sum_{a\in\mathcal{A}}(\pi(a|s) - \pi_t(a|s))^2 \tag{17}$$

$$= \arg\max_\pi\langle Q_{\pi_t}, \pi\rangle_{d_\pi} + \frac{1}{\alpha}\|\pi - \pi_t\|^2_{2,d_{\pi_t}} \tag{18}$$

which is a first-order Taylor expansion of J w.r.t. the policy's functional representation π (see Bhandari and Russo [2021, 2019])

$$J(\pi') = J(\pi) + \langle\nabla J(\pi), \pi' - \pi\rangle + \mathcal{O}(\|\pi' - \pi\|^2) \tag{19}$$

$$= J(\pi) + \langle Q_\pi, \pi' - \pi\rangle_{d_\pi} + \mathcal{O}(\|\pi' - \pi\|^2) \tag{20}$$

With per state decoupling, for specific values of α this yields a per state projection on the decoupled probability simplex

$$\pi_{t+1} = \mathcal{P}^\Pi_{[d_{\pi_t}]}\pi_{t+2} = \arg\max_{\pi\in\Pi}\|\pi - \pi_{t+2}\|^2_{2,d_{\pi_t}} \text{ with } \pi_{t+2} = \pi_t + \alpha Q_{\pi_t} \tag{21}$$

with $\|\cdot\|^2_{2,d_{\pi_t}}$ the weighted L_2-norm.

Mirror descent (MD) Mirror descent adapts to the geometry of the probability simplex by using a non-Euclidean regularizer. The specific regularizer used in RL is the entropy function $H(\pi) \equiv \pi \log \pi$, such that the resulting mirror map is the $\log$ function. The regularizer decouples across the state space and captures the curvature induced by the constraint of policies lying on the policy simplex via the softmax policy transform.

Starting with some policy $\pi_t \in \Pi_\Theta$, an iteration of mirror descent with step size α updates to the solution of a regularized problem

$$\pi_{t+1} = \arg\max_{\pi \in \Pi} \langle \nabla J(\pi_t), \pi \rangle + \frac{1}{\alpha} \sum_{s \in \mathcal{S}} d_{\pi_t}(s) \, \mathrm{KL}(\pi(s), \pi_t(s)) \tag{22}$$

$$= \arg\max_{\pi \in \Pi} \langle Q_{\pi_t}, \pi \rangle_{d_{\pi_t}} + \frac{1}{\alpha} \mathrm{KL}_{[d_{\pi_t}]}(\pi, \pi_t) \tag{23}$$

which is known to be the exponentiated gradient ascent update $\pi_{t+1} = \frac{\pi_t \exp \alpha Q_{\pi_t}}{\sum_a \pi(a|\cdot) \exp \alpha Q_{\pi_t}(\cdot, a)}$ (obtained using the Lagrange approach, see Bubeck [2015]).

Using state decoupling, for specific values of α we may also write MD as a projection using the corresponding Bregman divergence for the mirror map $\nabla_\pi H(\pi)$ (cf. Bubeck [2015])

$$\pi_{t+1} = \mathcal{P}^{\Pi, H}_{[d_{\pi_t}]} \pi_{t+2} = \arg\max_{\pi \in \Pi} \mathrm{KL}_{[d_\pi]}(\pi, \pi_{t+2}) \text{ with} \tag{24}$$

$$\log \pi_{t+2} = \log \pi_t + \alpha Q_{\pi_t} - \log \sum_a \pi(a|\cdot) \exp \alpha Q_{\pi_t}(\cdot, a) \tag{25}$$

Policy parametrization For parametric policy classes the search written over policies, translates into similar versions of the linear objective, except over policy parameters. Since the class of softmax policies can approximate stochastic policies to arbitrary precision, this is nearly (we can only come infinitesimally close to an optimal policy) the same as optimizing over the class Π.

Natural policy gradients (NPG) The natural policy gradient (NPG) of Kakade [2001] applied to the softmax parameterization is actually an instance of mirror descent for the entropy-based regularizer H.

Natural policy gradient is usually described as steepest descent in a variable metric defined by the Fisher information matrix induced by the current policy [Kakade, 2001, Agarwal et al., 2019]

$$\theta_{t+1} = \theta_t + \alpha \mathbf{F}_\rho(\theta_t)^\dagger \nabla_{\theta_t} J(\pi_{\theta_t}) \tag{26}$$

$$\mathbf{F}_\rho(\theta_t) = \mathbb{E}_{S \sim d_{\pi_{\theta_t}}, A \sim \pi_{\theta_t}} \left[\nabla_{\theta_t} \log \pi_{\theta_t} \nabla_{\theta_t} \log \pi_{\theta_t}^\top \right] \tag{27}$$

and is equivalent to mirror descent under some conditions [Raskutti and Mukherjee, 2014]. Cf. Bhandari and Russo [2021], Li et al. [2021], the aforementioned base MD and NPG updates are closely related to the practical instantiations in TRPO [Schulman et al., 2015], PPO [Schulman et al., 2017], MPO [Abdolmaleki et al., 2018], MDPO [Tomar et al., 2020]. All these algorithic instantiations use approximations for the gradient direction.

B.1.3 Actor-critic methods

Generally, in RL, an agent only has access to partial evaluations of the gradient $\nabla_\pi J(\pi)$, and commonly these involve some sort of internal representation of the action-value function $Q_t \approx Q_{\pi_t}$.

Natural actor-critic. MD with an estimated critic. Consider a parameterized softmax policy class $\pi_\theta \in \Pi_\Theta$, with parameter vector θ, and $Q_\eta \in \mathcal{F}_\eta$, with parameter vector η, s.t. For the softmax policy class, this will be $\log \pi_\theta$, for $\Pi_\Theta = \left\{ \pi_\theta \middle| \pi_\theta(s, a) = \frac{\exp f_\theta(s,a)}{\sum_{a' \in \mathcal{A}} \exp f_\theta(s, a')} \forall s \in \mathcal{S}, a \in \mathcal{A}, \theta \in \mathbb{R}^m \right\}$, with f_θ a differentiable function, either tabular $f_\theta(s,a) = \theta_{s,a}$, log-linear $f_\theta(s, a) = \phi(s, a)^\top \theta$, with ϕ a feature representation, or neural (f_θ-a neural network) parametrizations [Agarwal et al., 2019].

Written as a proximal policy improvement operator, at iteration t, starting with some policy $\pi_t \equiv \pi_{\theta_t}$. the next policy is the solution to the regularized optimization problem

$$\pi_{\theta_{t+1}} = \arg \max_{\pi_\theta \in \Pi_\Theta} \langle Q_{\eta_t}, \pi_\theta \rangle_{d_{\pi_t}} - \frac{1}{\alpha} \mathrm{KL}_{[d_{\pi_t}]}(\pi_\theta, \pi_t) \quad \textit{(optimistic improvement \& projection)} \tag{28}$$

with α a (possibly time-dependent) step size.

Using the connection between the NPG update rule with the notion of compatible function approximation [Sutton et al., 1999], as formalized in [Kakade, 2001], we may try to approximate the functional gradient using η

$$\mathbf{F}_\rho(\theta)^\dagger \nabla_\theta J(\pi_\theta) = \frac{\eta}{1-\gamma} \tag{29}$$

where η are parameters of an advantage function A_η—which is the solution to the projection of A_{π_θ} on the dual gradient space of π, the space spanned by the particular feature representation that uses $\phi_t \equiv \nabla_\theta \log \pi_{\theta_t}$ as (centered) features

$$\eta_t = \arg\min_\eta \mathbb{E}_{S\sim d_{\pi_{\theta_t}}, A\sim\pi_{\theta_t}}[(\eta^\top \phi_t(S,A) - \mathrm{A}_{\pi_{\theta_t}}(S,A))^2] \tag{30}$$

Similarly there is an equivalent version for Q-NPG considering possibly (un-centered) features ($\phi_{s,a}$, for $f_\theta(s,a) = \phi_{s,a}^\top \theta$) and projecting

$$\eta_t = \arg\min_\eta \mathbb{E}_{S\sim d_{\pi_{\theta_t}}, A\sim\pi_{\theta_t}}[(\eta^\top \phi_t(S,A) - Q_{\pi_{\theta_t}}(S,A))^2] \tag{31}$$

For both of them we can now replace the NPG parameter update with

$$\theta_{t+1} = \theta_t + \alpha\eta_t \tag{32}$$

B.1.4 Forward search

Multi-step policy iteration The single-step based policy improvement used in the aforementioned algorithms, e.g., policy iteration, approximate PI, actor-critic methods, and its practical algorithmic implementations, is not necessarily the optimal choice. It has been empirically demonstrated in RL algorithms based on Monte-Carlo Tree Search (MCTS)[Browne et al., 2012] (e.g., Schrittwieser et al. [2019], Schmidhuber [1987]) or Model Predictive Control (MPC), that multiple-step greedy policies can perform conspicuously better. Generalizations of the single-step greedy policy improvement include (i) h-step greedy policies, and (ii) κ–greedy policies. The former output the first optimal action out of a sequence of actions, solving a non-stationary h-horizon control problem:

$$\pi(s) \in \arg\max_{\pi_0} \max_{\pi_1,\dots\pi_{h-1}} \mathbb{E}^{\pi_0,\dots\pi_{h-1}}\left[\sum_{t=0}^{h-1} \gamma^t r(S_t, \pi_t(S_t)) + \gamma^h V(S_h) | S_0 = s\right] \tag{33}$$

equivalently described in operator notation as $\pi \in \mathcal{G}(\mathcal{T}^{h-1}V) \equiv \{\pi | \mathcal{T}_\pi T^{h-1} V \geq \mathcal{T}^h V\}$. A κ-greedy policy interpolates over all geometrically κ-weighted h-greedy policies $\pi \in \mathcal{G}(\mathcal{T}^\kappa V) \equiv \{\pi | \mathcal{T}_\pi^\kappa V \geq \mathcal{T}^\kappa V, \mathcal{T}_\pi^\kappa \equiv (1-\kappa)\sum_{h=0}^\infty \kappa^h \mathcal{T}_\pi^{h+1}\}$.

Multi-step soft policy iteration Efroni et al. [2018] shows that when using soft updates with $h > 1$

$$\pi_{t+1} = (1-\alpha)\pi_t + \alpha\pi_{t+1}^+, \pi_{t+1}^+ \in \mathcal{G}(\mathcal{T}^{h-1}V) \equiv \{\pi | \mathcal{T}_\pi T^{h-1} V \geq \mathcal{T}^h V\} \tag{34}$$

policy improvement is guaranteed only for $\alpha = 1$, and when using

$$\pi_{t+1} = (1-\alpha)\pi_t + \alpha\pi_{t+1}^+, \pi_{t+1}^+ \in \mathcal{G}(\mathcal{T}^\kappa V) \equiv \{\pi | \mathcal{T}_\pi^\kappa V \geq \mathcal{T}^\kappa V, \mathcal{T}_\pi^\kappa \equiv (1-\kappa)\sum_{h=0}^\infty \kappa^h \mathcal{T}_\pi^{h+1}\} \tag{35}$$

policy improvement is guaranteed only for $\alpha \in [\kappa, 1]$. This result appears in Efroni et al. [2018], and a more general version in Konda and Borkar [1999].

Tree search Notable examples of practical algorithms with empirical success that perform multi-step greedy policy improvement are AlphaGo and Alpha-Go-Zero [Silver et al., 2016a, 2017, 2016b], MuZero [Schrittwieser et al., 2019]. There, an approximate online version of multiple-step greedy improvement is implemented via Monte Carlo Tree Search (MCTS) [Browne et al., 2012]. In particular, Grill et al. [2020] shows that the tree search procedure implemented by AlphaZero is an approximation of the regularized optimization problem

$$\pi_{\theta_{t+1}} = \arg\max_{\pi_\theta \in \Pi_\Theta} \langle Q_t^h, \pi\rangle_{d_{\pi_t}} - \frac{1}{\alpha_t}\mathrm{KL}_{[d_{\pi_t}]}(\pi_\theta, \pi_t) \quad \textit{(optimistic improvement \& projection)} \tag{36}$$

with Q_t^h—the search Q-values, i.e., those estimated by the search algorithm that approximates $\mathcal{T}^h Q_{w_t}$ with stochastic sampling of trajectories in a tree up to a horizon h, and bootstrapping on a Q-fn estimator at the leaves. For a full description of the algorithm, refer to Silver et al. [2017]. The step size α_t captures the exploration strategy, and decreases the regularization based on the number of simulations.

B.1.5 Meta-learning

Optimistic meta-gradients Meta-gradient algorithms further relax the optimistic policy improvement step to a parametric update rule $\pi_{\theta_{t+1}} \equiv \varphi_{\pi_{\theta_t}}(\eta_t)$, e.g., $\theta_{t+1} = \theta_t + g_{\eta_t}$, when limited to a functional class of parametric GA update rules $g_\eta \in \mathcal{F}_\eta$. These algorithms implement adaptivity in a practical way, they project policy targets π_{t+2} ahead of $\pi_{\theta_{t+1}}$

$$g_{\eta_{t+1}} = \arg\min_{g_\eta \in \mathcal{F}_\eta} \mathrm{KL}_{[d_{\pi_{t+1}}]}(\pi_{\theta_{t+1}}, \pi_{t+2}) \qquad \textit{(hindsight adaptation \& projection)} \tag{37}$$

The targets can be parametric $\pi_{t+2} \equiv \pi_{\theta_{t+2}}$, initialized from $\theta_{t+1}^{(0)} = \theta_{t+1}$, and evolving for h step further ahead of θ_{t+1}, s.t. $\theta_{t+1}^{(k+1)} = \theta_{t+1}^k + g_t^k, \forall k \leq h$, with g_t^k representing predictions used by the bootstrapped targets–Alternatively, targets may be non-parameteric, e.g., $\pi_{t+2} \propto \pi_{\theta_{t+1}} \exp(Q_{t+1} - Q_t)$, e.g., if $Q_t^+ = \mathcal{T}_{\pi_{t+1}} Q_{\eta_t}$ then $\pi_{t+2} \propto \pi_{\theta_{t+1}} \exp(\mathcal{T}_{\pi_{t+1}} Q_{\eta_t} - Q_{\eta_t}) = \pi_{\theta_{t+1}}$—capturing the advantage of using the hypothesis $\pi_{\theta_{t+1}}$.

B.1.6 Optimism in online convex optimization

One way to design and analyze iterative optimization methods is through online linear optimization (OLO) algorithms.

Online learning Policy optimization through the lens of online learning [Hazan, 2017] means treating the policy optimization algorithm as the learner in online learning and each intermediate policy that it produces as an online decision. The following steps recast the iterative process of policy optimization into a standard online learning setup: (i) at iteration t the learner plays a decision $\pi_t \in \Pi$, (ii) the environment responds with feedback on the decision π_t, and the process repeats. The iteration t might be different than the timestep of the environment. Generally, it is assumed that the learner receives an unbiased stochastic approximation as a response, whereas that is not always the case for RL agents, using bootstrapping in their policy gradient estimation with a learned value function.

For an agent it is important to minimize the **regret** after T iterations

$$\mathrm{Reg}_T \equiv \sum_{t=0}^{T-1} \left(J(\pi^*) - J(\pi_t)\right) \tag{38}$$

The goal of **optimistic online learning** algorithms [Rakhlin and Sridharan, 2013a,b, 2014] is obtain better performance, and thus guaranteed lower regret, when playing against "easy" (i.e., predictable) sequences of online learning problems, where past information can be leveraged to improve on the decision at each iteration.

Predictability An important property of the above online learning problems is that they are not completely adversarial. In RL, the policy's true performance objective cannot be truly adversarial, as the same dynamics and cost functions are used across different iterations. In an idealized case where the true dynamics and cost functions are exactly known, using the policy returned from a model-based RL algorithm would incur zero regret, since only the interactions with the real MDP environment, not the model, are considered in the regret minimization problem formulation. The main idea is to use (imperfect) predictive models, such as off-policy gradients and simulated gradients, to improve policy learning.

B.1.7 Online learning algorithms

We now summarize two generalizations of the well-known core algorithms of online optimization for predictable sequences, cf. Joulani et al. [2020]: (i) a couple variants of optimistic mirror descent [Chiang et al., 2012, Rakhlin and Sridharan, 2013a,b, Chiang et al., 2012], including extragradient descent (Korpelevich [1976], and mirror-prox [Nemirovski, 2004, Juditsky et al., 2011], and (ii) adaptive optimistic follow-the-regularized-leader (AO-FTRL) [Rakhlin and Sridharan, 2013a, 2014, Mohri and Yang, 2016].

Optimistic mirror descent (OMD). Extragradient methods Starting with some previous iterate $x_t \in \mathcal{X}$, an OMD [Joulani et al., 2020] learner x uses a prediction $\tilde{g}_{t+1} \in \mathcal{X}^*$ ($\mathcal{X}^*$—dual space of $\mathcal{X}$) to minimize the regret on its convex loss function $f : \mathcal{X} \to \mathbb{R}$ against an optimal comparator $x^* \in \mathcal{X}$ with

$$x_{t+1} = \arg\min_{x \in \mathcal{X}} \langle g_t + \tilde{g}_{t+1} - \tilde{g}_t, x\rangle + \mathcal{B}_\Omega(x, x_t) \tag{39}$$

with $\tilde{g}_{t+1} \approx \nabla f(x_{t+1})$ optimistic gradient prediction, and $g_t \equiv \nabla f(x_t)$ true gradient feedback, $\mathcal{B}_\Omega$ a Bregman divergence with mirror map Ω.

Extragradient methods consider two-step update rules for the same objective using an intermediary sequence $\tilde{x}$

$$\tilde{x}_{t+1} = \arg\min_{x \in \mathcal{X}} \langle \tilde{g}_{t+1}, x \rangle + \mathcal{B}_\Omega(x, x_t) \tag{40}$$

$$x_{t+1} = \arg\min_{\pi \in \Pi} \langle g^+_{t+1}, x \rangle + \mathcal{B}_\Omega(x, x_t) \tag{41}$$

with $\tilde{g}_{t+1} \approx \nabla f(x_{t+1})$ a gradient prediction, and $g^+_{t+1} \equiv \nabla f(\tilde{x}_{t+1})$ the true gradient direction, but for the intermediary optimistic iterate $\tilde{x}_{t+1}$.

Adaptive optimistic follow-the-regularized-leader (AO-FTRL) A learner using AO-FTRL updates x using

$$x_{t+1} = \arg\min_{x \in \mathcal{X}} \langle g_{0:t} + \tilde{g}_{t+1}, x \rangle + \omega_{1:t-1}(x) \tag{42}$$

where $g_{0:t} = \sum_{j=0}^{t} g_j$ are true gradients, $\tilde{g}_{t+1}$ is the optimistic part of the update, a prediction of the gradient before it is received, and $\omega_{0:t}(x) = \sum_{j=0}^{t} \omega_j(x)$ represent the "proximal" part of this adaptive regularization (cf. Joulani et al. [2020]), counterparts of the Bregman divergence we have for MD updates that regularizes iterates to maintain proximity.

B.1.8 Policy optimization with online learning algorithms

Cheng et al. [2018] follows the extragradient approach for policy optimization

$$\pi_{t+2} = \arg\max_{\pi \in \Pi} \langle Q_t, \pi \rangle + \mathrm{KL}(\pi, \pi_t) \tag{43}$$

$$\pi_{t+1} = \arg\max_{\pi \in \Pi} \langle Q_{\pi_t}, \pi \rangle - \mathrm{KL}(\pi, \pi_t) \tag{44}$$

but changes the second sequence to start from the intermediary sequence and add just a correction

$$\pi_{t+2} = \arg\max_{\pi \in \Pi} \langle Q_t, \pi \rangle - \mathrm{KL}(\pi, \pi_t) \tag{45}$$

$$\pi_{t+1} = \arg\max_{\pi \in \Pi} \langle Q_{\pi_t} - Q_t, \pi \rangle - \mathrm{KL}(\pi, \pi_{t+2}) \tag{46}$$

This approach uses π_{t+2} as the optimistic prediction, and π_{t+1} as the hindsight corrected prediction—a policy optimal in hindsight w.r.t. the average of all previous Q-functions rather than just the most recent one. But it needs an additional model for the value functions Q_t, and another learning algorithm to adapt Q_t to the Q_{t+1}. Additionally, an agent does not generally have access to Q_{π_t}, but only partial evaluations.

Hao et al. [2020] also designs an adaptive optimistic algorithm based on AO-FTRL, which updates

$$\pi_{t+1} = \arg\max_{\pi \in \Pi} \left\langle \left(\sum_{j=0}^{t} Q_j + \hat{Q}_{t+1} \right), \pi \right\rangle - \alpha_t \omega(\pi) \tag{47}$$

with ω—a regularizer, and with $Q_j \approx Q_{\pi_j}, \forall j \le t$ predictions for the true gradients, and $\hat{Q}_{t+1} \approx Q_{\pi_{t+1}}$ is also a prediction for the gradient of the next policy, which uses the previous predictions $Q_{\pi_j}, \forall j \le t$ to compute it. The authors also propose an adaptive method for learning α_t that uses gradient errors of $Q_j, \forall j \le t$. Averaging value functions has also been explored by Vieillard et al. [2020b] and Vieillard et al. [2020a].

C Empirical analysis details

Table 2: Notation

t	iterations/timesteps
T	number of iterations
n	rollout length
μ	buffer
$\mathcal{M}$	meta-buffer
w	standard critic (Q-fn Q_w) parameters
η	meta parameters of meta-learner (g_η or Q_η)
ν	step size for meta-learner's parameters η (Q-fn Q_η)
ζ	step size for standard critic's parameters w (Q-fn Q_w)
ξ	step size for the policy learner's parameters θ (π_θ)
h	lookahead horizon
Q_{t+h}	search Q-values up to lookahead horizon h (tree depth)

C.1 Algorithms

Algorithm 3 Policy gradient

```
1: Init: params $\theta_0$, buffer $\mu = [()]$
2: for $t \in 0..T$ iterations do
3:   Every $n$ steps using a rollout $\mu \leftarrow (S_t, A_t, R_t, S_{t+1} \dots S_{t+n}) \sim \pi_{\theta_t}$
4:       Update policy learner $\pi_{\theta_{t+1}}$ cf. Eq.48
5: end for
```

Policy gradients Algorithm 3 describes a standard PG algorithm (cf. [Williams, 1992]) with an expert oracle critic Q_{π_θ}, for the policy evaluation of π_θ. The standard policy gradient update is

$$\theta_{t+1} = \theta_t + \xi \frac{1}{n} \sum_{i=t}^{t+n} \nabla_{\theta_t} \log \pi_{\theta_t}(A_i|S_i) \left(Q_{\pi_{\theta_t}}(S_i, A_i) - \mathbb{E}_{\pi_{\theta_t}}[Q_{\pi_{\theta_t}}(S_i, \cdot)] \right) \tag{48}$$

Algorithm 4 Actor-critic

```
1: Init: params $(\theta_0, w_0)$, buffer $\mu = [()]$
2: for $t \in 0..T$ iterations do
3:   Every $n$ steps using a rollout $\mu \leftarrow (S_t, A_t, R_t, S_{t+1} \dots S_{t+n}) \sim \pi_{\theta_t}$
4:       Update critic $Q_{w_{t+1}}$ cf. Eq.50 and policy learner $\pi_{\theta_{t+1}}$ cf. Eq.49
5: end for
```

Actor-critic Algorithm 4 describes a standard AC algorithm (cf. [Sutton et al., 1999]) with an estimated critic Q_w, for the policy evaluation of π_θ. The policy updates

$$\theta_{t+1} = \theta_t + \xi \frac{1}{n} \sum_{i=t}^{t+n} \nabla_{\theta_t} \log \pi_{\theta_t}(A_i|S_i) \left(Q_{w_t}(S_i, A_i) - \mathbb{E}_{\pi_{\theta_t}}[Q_{w_t}(S_i, \cdot)] \right) \tag{49}$$

and the critic's update using TD(0) learning, writes

$$w_{t+1} = w_t - \zeta \frac{1}{n} \sum_{i=t}^{t+n} \left(R_i + \gamma E_{\pi_t}[Q_{w_t}(S_{i+1}, \cdot)] - Q_{w_t}(S_i, A_i) \right) \nabla_{w_t} Q_{w_t}(S_i, A_i) \tag{50}$$

Algorithm 5 Policy gradients with forward search

```
1: Init: params $(\theta_0, w_0)$, buffer $\mu = [()]$
2: for $t \in 0..T$ iterations do
3:     Every $n$ steps using a rollout $\mu \leftarrow (S_t, A_t, R_t, S_{t+1} \dots S_{t+n}) \sim \pi_{\theta_t}$
4:         Generate search Q-values $Q_{t+h}$ up to lookahead horizon $h$ with
5:             (i) $Q_{t+h} = \mathcal{T}^h_{\pi_t} Q_{w_t}$
6:             (ii) $Q_{t+h} = \mathcal{T}^h Q_{w_t}$
7:         Update critic $Q_{w_{t+1}}$ cf. Eq.52 and policy learner $\pi_{\theta_{t+1}}$ cf. Eq.51, using $Q_{t+h}$
8: end for
```

Forward search with a model Algorithm 5 describes an AC algorithm with h-step lookahead search in the gradient critic

$$\theta_{t+1} = \theta_t + \xi \frac{1}{n} \sum_{i=0}^{n} \nabla_{\theta_t} \log \pi_{\theta_t}(A_i|S_i) \left(Q_{t+h}(S_i, A_i) - \mathbb{E}_{\pi_t}[Q_{t+h}(S_i, \cdot)]\right) \tag{51}$$

where Q_{t+h} is either (i) $Q_{t+h} = \mathcal{T}^h Q_{w_t}$ or (ii) $Q_{t+h} = \mathcal{T}^h Q_{w_t}$, depending on the experimental setup, and the critic is updated toward the search Q-values

$$w_{t+1} = w_t - \zeta \frac{1}{n} \sum_{i=0}^{n} \nabla_{w_t} Q_{w_t}(S_i, A_i) \left(R_i + \gamma E_{\pi_t}[Q_{t+h}(S_{i+1}, \cdot)] - Q_{w_t}(S_i, A_i)\right) \tag{52}$$

Algorithm 6 Optimistic policy gradients with policy targets computed from expert hints

```
1: Init: params $(\theta_0, \eta_0)$, buffer $\mu = [()]$, meta-buffer $\mathcal{M} = [\mu, ..]$
2: for $t \in 0..T$ iterations do
3:     Every $n$ steps using a rollout $\mu_t \leftarrow (S_t, A_t, R_t, S_{t+1} \dots S_{t+n}) \sim \pi_{\theta_t}$
4:         Predict $u_{\eta_{t-1}} = \varphi(\eta_{t-1}, Q_{\pi_{\theta_{t+1}}})$
5:         Update learner with using optimistic prediction $u_{\eta_{t-1}}$ using Eq. 53
6:         Every $h$ steps using experience stored in the meta-buffer $\mathcal{M} \leftarrow (\mu_t, \dots \mu_{t+h})$
7:             Compute policy targets $\pi_{\theta_{t+2}}$ cf. Eq. 54 or $\pi_{t+2}$ cf. Eq. 55
8:             Update meta-learner $u_{\eta_t}$ cf. Eq. 56
9: end for
```

Optimistic policy gradients with expert targets (hints) Algorithm 6 describes a meta-gradient based algorithm for learning *optimistic policy gradients* by supervised learning from policy targets computed with complete hints (Q_π). The meta-update used updates

$$\theta_{t+1} = \theta_t + \xi u_{\eta_{t-1}} \tag{53}$$

where $u_{\eta_t} = \frac{1}{n} \sum_{i=0}^{n} \nabla_{\theta_t} \log \pi_{\theta_t}(A_i|S_i) \left((U_{\eta_t}(S_i, A_i) - \mathbb{E}_{\pi_{\theta_t}}[(U_{\eta_t}(S_i, A_i)]\right)$ The policy targets are (i) *parametric policies* obtained at iteration t by starting from the parameters θ_{t+1} ($\tilde{\theta}^0_{t+1} = \theta_{t+1}$) and executing h parameter updates with data from successive batches of rollouts $\mu_{t+1:t+h}$ sampled from the meta-buffer $\mathcal{M}$

$$\theta^{j+1}_{t+2} = \theta^j_{t+2} + \xi \hat{g}^j_{t+2} \tag{54}$$

with $\hat{g}^j_{t+2} = \frac{1}{n} \sum_{i=0}^{n} \nabla_{\theta_t} \log \pi_{\theta^j_t}(A_i|S_i) \left((Q_{\pi_{t+1}}(S_i, A_i) - \mathbb{E}_{\pi_{\theta_t}}[(Q_{\pi_{t+1}}(S_i, A_i)]\right)$. After h steps the resulting target parameters $\theta_{t+2} \equiv \theta^h_{t+2}$, and yield the target policy $\pi_{\theta_{t+2}}$.

The other choice we experiment with is to use a target constructed with (ii) *geometric updates* for one (or more) steps ahead, similarly to tree-search policy improvement procedures. The targets are initialized with $\pi^0_{t+1} = \pi_{\theta_{t+1}}$ and execute one (or more) steps of policy improvement

$$\pi^{j+1}_{t+2} \propto \pi^j_{t+2} \exp \alpha Q_{\pi_{t+1}} \tag{55}$$

yielding the non-parametric policy target $\pi_{t+2} \equiv \pi^{j+1}_{t+2}$. Setting $\alpha \to \infty$ in Eq.55, if the predictions are given, or can be computed with the help of the simulator model, we obtain an update similar to the multi-step greedy operator $\mathcal{T}^h$ used in forward search.

The next parameter vector η_t for the gradient u_η is distilled via meta-gradient learning by projecting the expert policy target π_{t+2} (or $\pi_{\theta_{t+2}}$) using the data samples from $\mu_t, \dots \mu_{t+h}$ from $\mathcal{M}$ and the surrogate objective

$$\eta_{t+1} = \eta_t - \nu \frac{1}{h} \sum_{j=t}^{t+h} \nabla_{\eta_t} \mathrm{KL}(\pi_{\theta_{t+1}}(S_j), \pi_{t+2}(S_j)) \tag{56}$$

Algorithm 7 Optimistic policy gradients with target predictions

```
1:  Init: params $(\theta_0, w_0, \eta_0)$, buffer $\mu = [()]$, meta-buffer $\mathcal{M} = [\mu, ..]$
2:  for $t \in 0..T$ iterations do
3:    Every $n$ steps using a rollout $\mu_t \leftarrow (S_t, A_t, R_t, S_{t+1} \dots S_{t+n}) \sim \pi_{\theta_t}$
4:        Predict $u_{\eta_t}$
5:        Update learner with optimistic prediction $u_{\eta_t}$ using Eq. 53
6:        Update $Q_{w_{t+1}}$ cf. Eq.50
7:        Every $h$ steps using experience stored in the meta-buffer $\mathcal{M} \leftarrow (\mu_t, \dots \mu_{t+h})$
8:            Compute policy targets $\pi_{\theta_{t+2}}$ cf. Eq. ?? or $\pi_{t+2}$ cf. Eq. ??
9:            Update meta-learner $u_{\eta_{t+1}}$ cf. Eq. 56
10: end for
```

Optimistic policy gradients with target predictions Algorithm 7 describes a meta-gradient based algorithm for learning *optimistic policy gradients*, by self-supervision from target predictions (learned estimators). The targets we use are (i) *parametric*, computed at iteration t, similarly to the previous paragraph (Eq. 54, except we now replace the hint that was using complete evaluations, with a partial evaluation. We also experiment with the (ii) *non-parametric target* that takes *geometric* steps, similarly to tree-search policy improvement procedure, where the hint from Eq.55, uses complete evaluations.

C.2 Experimental setup

Environment details All empirical studies are performed on the same discrete navigation task from Sutton and Barto [2018], illustrated in Fig. 3. "G" marks the position of the goal and the end of an episode. "S" denotes the starting state to which the agent is reset at the end of the episode. The state space size is 48, $\gamma = 0.99$. There are 4 actions that can transition the agent to each one of the adjacent states. Reward is 1 at the goal, and zero everywhere else. Episodes terminate and restart from the initial state upon reaching the goal.

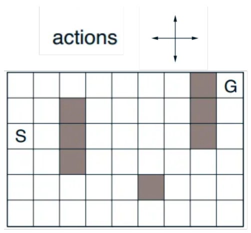


Figure 3: Maze Navigation: illustration of the MDP used in the empirical studies

Protocol All empirical studies report the regret of policy performance every step, and at every episode $J(\pi^*) - J(\pi_t)$, for a maximum number of 500 episodes. Hyperparameter sensitivity plots show the cumulative regret per total number of steps of experience cumulated in 500 episodes, $\sum_t J(\pi^*) - J(\pi_t)$. This quantity captures the sample efficiency in terms of number of steps of interaction required.

Algorithmic implementation details Meta-gradient based algorithms keep parametric representations of the gradient fields via a parametric advantage $\mathrm{A}_\eta(s, a) = U_\eta(s, a) - E_\pi[U_\eta(s, A)], \forall s, a$, s.t. a learned gradient update consists of a parametric gradient step on the loss

$$\theta_{t+1} = \theta_t + \nabla \mathcal{L}(\theta; \mu_t)\Big|_{\theta=\theta_t} \tag{57}$$

$$\mathcal{L}(\theta; \mu_t) = \frac{1}{n} \sum_{i=t}^{t+n} \log \pi_\theta(A_i|S_i) \left(U_\eta(S_i, A_i) - \sum_a \pi_{\theta_t}(A_i|S_i) U_\eta(S_i, A_i) \right) \tag{58}$$

Policies use the standard softmax transform $\pi_\theta = \frac{\exp f_\theta(s,a)}{\sum_b \exp f_\theta(s,b)}$, with f_θ, the policy logits. In the experiments illustrated we use a tabular, one-hot representation of the state space as features, so f_θ is essentially θ. The same holds for the critic's parameter vector Q_w, and the meta-learner's parameter vector Q_η.

The experiments were written using JAX, Haiku, and Optax [Bradbury et al., 2018, Babuschkin et al., 2020, Hennigan et al., 2020].

Experimental details for the forward search experiment We used forward search with the environment true dynamics model up to horizon h, backing-up the current value estimate Q_{w_t} at the leaves. We distinguish between two settings: (i) using the previous policy π_t for *bootstrapping* in the tree-search back-up procedure, i.e. obtaining $Q_t^h = \mathcal{T}_{\pi_t}^h Q_{w_t}$ at the root of the tree; or (ii) using the greedification inside the tree to obtain $Q_t^h = \mathcal{T}^h Q_{w_t}$ at the root. Table 3 specifies the hyperparameters used for both of the aforementioned experimental settings. Results shown in the main text are averaged over 10 seeds and show the standard error over runs.

Table 3: Hyperparameters for optimism via forward search on the Maze Gridworld in Fig. 3

Hyperparameter	
ξ (policy step size)	0.5
ζ (Q-fn step size)	{0.01, 0.1, 0.5, 0.9}
h (lookahead horizon)	{0, 1, 2, 4, 8, 16}
n (rollout length)	2
policy/Q-fn optimiser	SGD

Experimental details for the meta-gradient experiments with expert target/hints For this experiment we used Algorithm 6 described in Sec. C.1, and the hyperaparameters in Table 4.

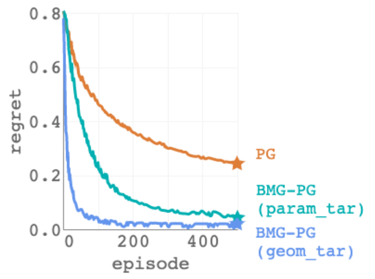
Table 4: Hyperparameters for optimism via meta-gradient learning with expert targets/hints on the Maze Gridworld in Fig. 3

Hyperparameter	
ξ (policy step size)	0.1 (training plots Fig.2-d) {0.1, 0.5} (sensitivity plots Fig.2-c)
ζ (Q-fn step size)	-
h (lookahead horizon)	1
α (step size π)	1
n (rollout length)	2
policy optimiser	SGD
meta-learner optimiser	Adam

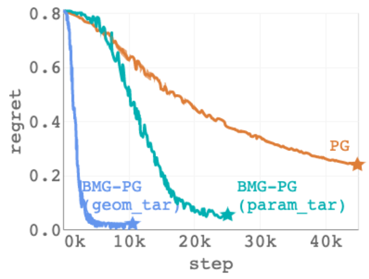
Experimental details for the meta-gradient experiments with target predictions/bootstrapping For this experiment we used Algorithm 7 described in Sec. C.1, and the hyperaparameters in Table 5.

Table 5: Hyperparameters for optimism via meta-gradient learning with target predictions/bootstrapping on the Maze Gridworld in Fig. 3

Hyperparameter	
ξ (policy step size)	0.5
ζ (Q-fn step size)	0.1 (training plots Fig.2-e) {0.1, 0.5} (sensitivity plots Fig.2-f)
h (lookahead horizon)	1
n (rollout length)	2
policy optimiser	SGD
meta-learner optimiser	Adam

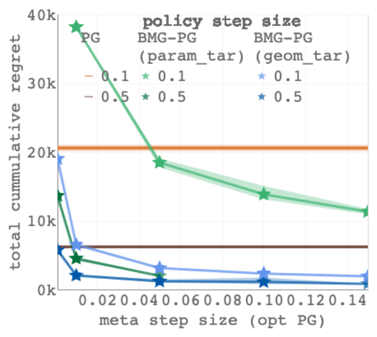


(a) expert targets, Adam, perf/episode

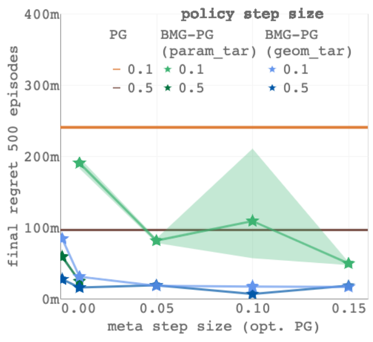


(b) expert targets, Adam, perf/step

Figure 4: Meta-learner uses Adam. Policy optimization with adaptive optimistic policy gradients. x-axis - (a) no of episodes, (b) no of steps. y-axis - regret $J(\pi^*) - J(\pi_t)$. Learning curves denote: the baseline - standard PG algorithm, *adaptive optimistic policy gradient learning algorithms* - with parametric target policies , functional non-parametric target policies, trained with meta-gradients from *expert targets*. Shades (wherever noticeable) denote standard error over different runs.



(a) Hyperparam. sensitivity / expert targets / Adam / total regret

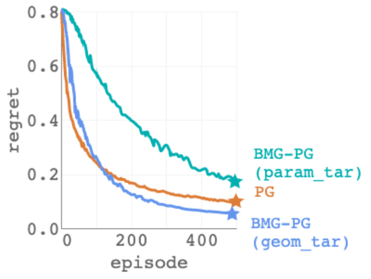


(b) Hyperparam. sensitivity / expert targets / Adam / final regret

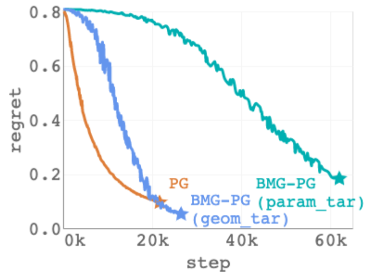
Figure 5: Meta-learner uses Adam. Hyper-parameter sensitivity curves for the meta-learning rate ν - x-axis, y-axis - total cummulative regret (a): $\sum_{i \leq t} J(\pi^*) - J(\pi_i)$, final regret (b): $J(\pi^*) - J(\pi_T)$; Learning curves show *adaptive optimistic policy gradient learning algorithms* - with parametric target policies , functional non-parametric target policies, trained with meta-gradients from *expert targets*. Different tones show the evolution of the meta-hyperparameter ν to those used in the inner learned optimization algorithm, i.e. the policy step size. Different straight lines denote the baseline—standard PG. Shades denote standard error over different runs.

C.3 Additional results & observations

Fig 4 shows learning curves, and Fig 5 hyperparameter sensitivity, for experiments with expert targets, when using Adam for the meta-optimization. Fig 6 (learning curves), and Fig 7 (hyperparameter sensitivity) illustrate results for when the meta-optimization uses SGD. The next set of figures show experiments with target predictions—for Adam Fig 8 (learning curves) and Fig 9 (hyperparameter sensitivity), and for SGD Fig 10 (learning curves) and Fig 11 (hyperparameter sensitivity).

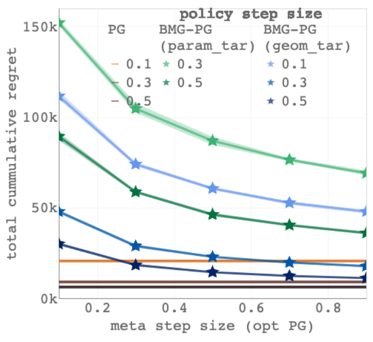


(a) expert targets, SGD, perf/episode

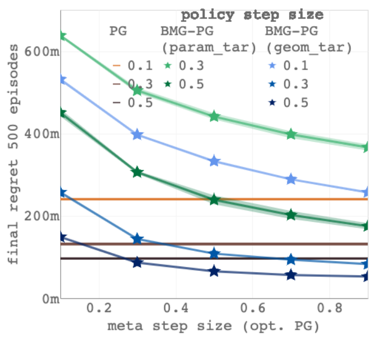


(b) expert targets, SGD, perf/step

Figure 6: Meta-learner uses SGD. Policy optimization with adaptive optimistic policy gradients. x-axis - (a) no of episodes, (b) no of steps. y-axis - regret $J(\pi^*) - J(\pi_t)$. Learning curves denote: the baseline - standard PG algorithm, *optimistic policy gradient learning algorithms* - with parametric target policies , functional non-parametric target policies, trained with meta-gradients from *expert targets*. Shades (wherever noticeable) denote standard error over different runs.

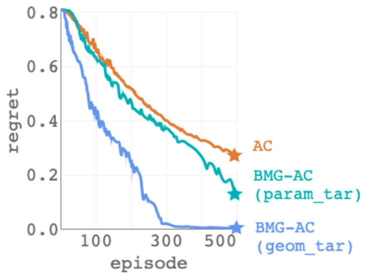


(a) expert targets, SGD, total regret

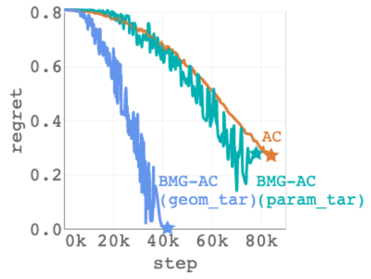


(b) expert targets, SGD, final regret

Figure 7: Meta-learner uses SGD. Hyper-parameter sensitivity curves for the meta-learning rate ν - x-axis. y-axis - total cumulative regret (a): $\sum_{i\leq t} J(\pi^*) - J(\pi_i)$, final regret (b): $J(\pi^*) - J(\pi_T)$. Learning curves show *adaptive optimistic policy gradient learning algorithms* - with parametric target policies, functional non-parametric target policies, trained with meta-gradients from *expert targets*. Different tones show the evolution of the meta-hyperparameter ν to those used in the inner learned optimization algorithm, i.e. the policy step size. Different straight lines denote the baseline—standard PG. Shades denote standard error over different runs.

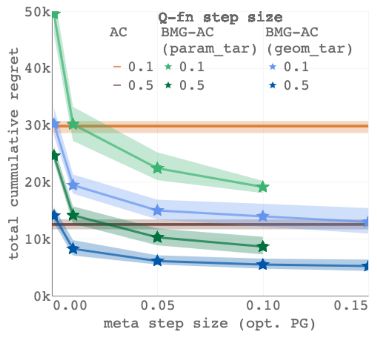


(a) target pred, Adam, perf/episode

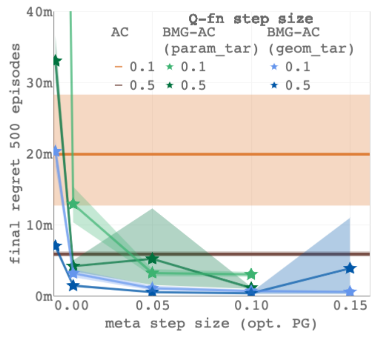


(b) target pred, Adam, perf/step

Figure 8: Meta-learner uses Adam. Policy optimization with adaptive optimistic policy gradients. x-axis - (a) no of episodes, (b) no of steps. y-axis - regret $J(\pi^*) - J(\pi_t)$. Learning curves denote: the baseline - standard AC algorithm, *optimistic policy gradient learning algorithms* - with parametric target policies , functional non-parametric target policies, trained with meta-gradients from *target predictions*. Shades (wherever noticeable) denote standard error over different runs.

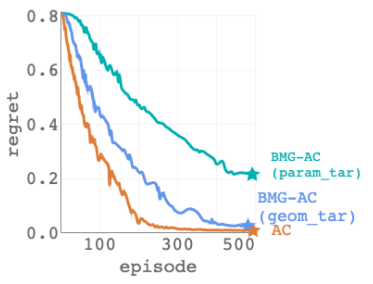


(a) target pred, Adam, total regret

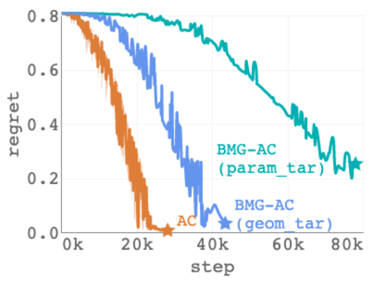


(b) target pred, Adam, final regret

Figure 9: Meta-learner uses Adam. Hyper-parameter sensitivity curves for the meta-learning rate ν - x-axis. y-axis - total cummulative regret (a): $\sum_{i \leq t} J(\pi^*) - J(\pi_i)$, final regret (b): $J(\pi^*) - J(\pi_T)$. Learning curves show *optimistic policy gradient learning algorithms* - with parametric target policies , functional non-parametric target policies, trained with meta-gradients from *target predictions*. Different tones show the evolution of the meta-hyperparameter ν to those used in the inner learned optimization algorithm, i.e. Q-fn step size. Different straight lines denote the baseline—standard AC. Shades denote standard error over different runs.

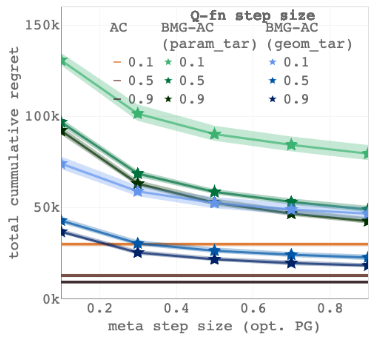


(a) target pred, SGD, perf/episode

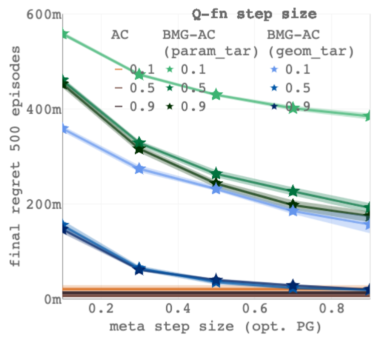


(b) target pred, SGD, perf/step

Figure 10: Meta-learner uses SGD. Policy optimization with optimistic policy gradients. x-axis - (a) no of episodes, (b) no of steps. y-axis - regret $J(\pi^*) - J(\pi_t)$. Learning curves denote: the baseline - standard AC algorithm, *optimistic policy gradient learning algorithms* - with parametric target policies, functional non-parametric target policies, trained with meta-gradients from *target predictions*. Shades (wherever noticeable) denote standard error over different runs.



(a) target pred, SGD, total regret



(b) target pred, SGD, final regret

Figure 11: Meta-learner uses SGD. Hyper-parameter sensitivity curves for the meta-learning rate ν - x-axis; y-axis - total cummulative regret (a): $\sum_{i \leq t} J(\pi^*) - J(\pi_i)$, final regret (b): $J(\pi^*) - J(\pi_T)$. Learning curves show *optimistic policy gradient learning algorithms* - with parametric target policies , functional non-parametric target policies, trained with meta-gradients from *target predictions*. Different tones show the evolution of the meta-hyperparameter ν to those used in the inner learned optimization algorithm, i.e. Q-fn step size. Different straight lines denote the baseline—standard AC. Shades denote standard error over different runs.

References

Y. Abbasi-Yadkori, P. Bartlett, K. Bhatia, N. Lazic, C. Szepesvari, and G. Weisz. POLITEX: Regret bounds for policy iteration using expert prediction. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3692–3702. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/lazic19a.html`.

A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. A. Riedmiller. Maximum a posteriori policy optimisation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=S1ANxQW0b`.

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. *CoRR*, abs/1908.00261, 2019. URL `http://arxiv.org/abs/1908.00261`.

C. Alfano and P. Rebeschini. Linear convergence for natural policy gradient with log-linear policy parametrization, 2023.

D. G. M. Anderson. Iterative procedures for nonlinear integral equations. *J. ACM*, 12(4):547–560, 1965. doi: 10.1145/321296.321305. URL `https://doi.org/10.1145/321296.321305`.

K. Asadi, R. Fakoor, O. Gottesman, M. L. Littman, and A. J. Smola. Deep q-network with proximal iteration. *CoRR*, abs/2112.05848, 2021. URL `https://arxiv.org/abs/2112.05848`.

I. Babuschkin, K. Baumli, A. Bell, S. Bhupatiraju, J. Bruce, P. Buchlovsky, D. Budden, T. Cai, A. Clark, I. Danihelka, A. Dedieu, C. Fantacci, J. Godwin, C. Jones, R. Hemsley, T. Hennigan, M. Hessel, S. Hou, S. Kapturowski, T. Keck, I. Kemaev, M. King, M. Kunesch, L. Martens, H. Merzic, V. Mikulik, T. Norman, G. Papamakarios, J. Quan, R. Ring, F. Ruiz, A. Sanchez, R. Schneider, E. Sezener, S. Spencer, S. Srinivasan, W. Stokowiec, L. Wang, G. Zhou, and F. Viola. The DeepMind JAX Ecosystem, 2020. URL `http://github.com/deepmind`.

D. P. Bertsekas. Approximate policy iteration: a survey and some new methods. *Journal of Control Theory and Applications*, 9(3):310–335, 2011. doi: 10.1007/s11768-011-1005-3. URL `https://doi.org/10.1007/s11768-011-1005-3`.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Number 3 in the Optimization and Neural Computation Series. Athena Scientific, Belmont, Mass, 1996.

J. Bhandari and D. Russo. Global optimality guarantees for policy gradient methods. *CoRR*, abs/1906.01786, 2019. URL `http://arxiv.org/abs/1906.01786`.

J. Bhandari and D. Russo. On the linear convergence of policy gradient methods for finite mdps. In A. Banerjee and K. Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2386–2394. PMLR, 13–15 Apr 2021. URL `https://proceedings.mlr.press/v130/bhandari21a.html`.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, et al. Jax: composable transformations of python+ numpy programs. 2018.

C. Browne, E. J. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. P. Liebana, S. Samothrakis, and S. Colton. A survey of monte carlo tree search methods. *IEEE Trans. Comput. Intell. AI Games*, 4(1):1–43, 2012. doi: 10.1109/TCIAIG.2012.2186810. URL `https://doi.org/10.1109/TCIAIG.2012.2186810`.

S. Bubeck. Convex optimization: Algorithms and complexity. *Found. Trends Mach. Learn.*, 8(3-4):231–357, 2015. doi: 10.1561/2200000050. URL `https://doi.org/10.1561/2200000050`.

S. Cen, C. Cheng, Y. Chen, Y. Wei, and Y. Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Oper. Res.*, 70(4):2563–2578, 2022. doi: 10.1287/opre.2021.2151. URL `https://doi.org/10.1287/opre.2021.2151`.

Y. Chandak, G. Theocharous, S. Shankar, M. White, S. Mahadevan, and P. S. Thomas. Optimizing for the future in non-stationary mdps. *CoRR*, abs/2005.08158, 2020. URL `https://arxiv.org/abs/2005.08158`.

Y. Chebotar, A. Molchanov, S. Bechtle, L. Righetti, F. Meier, and G. S. Sukhatme. Meta-learning via learned loss. *CoRR*, abs/1906.05374, 2019. URL `http://arxiv.org/abs/1906.05374`.

Z. Chen and S. Theja Maguluri. Sample complexity of policy-based methods under off-policy sampling and linear function approximation. In G. Camps-Valls, F. J. R. Ruiz, and I. Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 11195–11214. PMLR, 28–30 Mar 2022. URL `https://proceedings.mlr.press/v151/chen22i.html`.

C. Cheng, X. Yan, N. D. Ratliff, and B. Boots. Predictor-corrector policy optimization. *CoRR*, abs/1810.06509, 2018. URL `http://arxiv.org/abs/1810.06509`.

C.-K. Chiang, T. Yang, C.-J. Lee, M. Mahdavi, C.-J. Lu, R. Jin, and S. Zhu. Online optimization with gradual variations. In S. Mannor, N. Srebro, and R. C. Williamson, editors, *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23 of *Proceedings of Machine Learning Research*, pages 6.1–6.20, Edinburgh, Scotland, 25–27 Jun 2012. PMLR. URL `https://proceedings.mlr.press/v23/chiang12.html`.

Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016. URL `http://arxiv.org/abs/1611.02779`.

Y. Efroni, G. Dalal, B. Scherrer, and S. Mannor. Multiple-step greedy policies in online and approximate reinforcement learning. *CoRR*, abs/1805.07956, 2018. URL `http://arxiv.org/abs/1805.07956`.

L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu. IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures. *CoRR*, abs/1802.01561, 2018. URL `http://arxiv.org/abs/1802.01561`.

C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400, 2017. URL `http://arxiv.org/abs/1703.03400`.

S. Flennerhag, Y. Schroecker, T. Zahavy, H. van Hasselt, D. Silver, and S. Singh. Bootstrapped meta-learning. *CoRR*, abs/2109.04504, 2021. URL `https://arxiv.org/abs/2109.04504`.

M. Geist and B. Scherrer. Anderson acceleration for reinforcement learning. *CoRR*, abs/1809.09501, 2018. URL `http://arxiv.org/abs/1809.09501`.

V. Goyal and J. Grand-Clement. A first-order approach to accelerated value iteration, 2021.

J.-B. Grill, F. Altché, Y. Tang, T. Hubert, M. Valko, I. Antonoglou, and R. Munos. Monte-carlo tree search as regularized policy optimization, 2020. URL `https://arxiv.org/abs/2007.12509`.

A. Gupta, R. Mendonca, Y. Liu, P. Abbeel, and S. Levine. Meta-reinforcement learning of structured exploration strategies. *CoRR*, abs/1802.07245, 2018. URL `http://arxiv.org/abs/1802.07245`.

B. Hao, N. Lazic, Y. Abbasi-Yadkori, P. Joulani, and C. Szepesvári. Provably efficient adaptive approximate policy iteration. *CoRR*, abs/2002.03069, 2020. URL `https://arxiv.org/abs/2002.03069`.

E. Hazan. *Introduction to Online Convex Optimization*. Foundations and Trends in Optimization. Now, Boston, 2017. ISBN 978-1-68083-170-2. doi: 10.1561/2400000013.

T. Hennigan, T. Cai, T. Norman, and I. Babuschkin. Haiku: Sonnet for JAX, 2020. URL `http://github.com/deepmind/dm-haiku`.

M. Hessel, I. Danihelka, F. Viola, A. Guez, S. Schmitt, L. Sifre, T. Weber, D. Silver, and H. van Hasselt. Muesli: Combining improvements in policy optimization. *CoRR*, abs/2104.06159, 2021. URL `https://arxiv.org/abs/2104.06159`.

R. Houthooft, R. Y. Chen, P. Isola, B. C. Stadie, F. Wolski, J. Ho, and P. Abbeel. Evolved policy gradients. *CoRR*, abs/1802.04821, 2018. URL `http://arxiv.org/abs/1802.04821`.

P. Joulani, A. Raj, A. Gyorgy, and C. Szepesvari. A simpler approach to accelerated optimization: iterative averaging meets optimism. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4984–4993. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/joulani20a.html`.

A. Juditsky, A. S. Nemirovskii, and C. Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm, 2011.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, page 267–274, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1558608737.

S. M. Kakade. A natural policy gradient. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL `https://proceedings.neurips.cc/paper/2001/file/4b86abe48d358ecf194c56c69108433e-Paper.pdf`.

S. Khodadadian, P. R. Jhunjhunwala, S. M. Varma, and S. T. Maguluri. On the linear convergence of natural policy gradient algorithm. *CoRR*, abs/2105.01424, 2021. URL `https://arxiv.org/abs/2105.01424`.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

L. Kirsch, S. van Steenkiste, and J. Schmidhuber. Improving generalization in meta reinforcement learning using learned objectives. *CoRR*, abs/1910.04098, 2019. URL `http://arxiv.org/abs/1910.04098`.

V. R. Konda and V. S. Borkar. Actor-critic–type learning algorithms for markov decision processes. *SIAM Journal on Control and Optimization*, 38(1):94–123, 1999. doi: 10.1137/S036301299731669X. URL `https://doi.org/10.1137/S036301299731669X`.

G. M. Korpelevich. The extragradient method for finding saddle points and other problems. 1976.

G. Lan. Policy mirror descent for reinforcement learning: Linear convergence, new sampling complexity, and generalized problem classes, 2022.

N. Lazic, D. Yin, Y. Abbasi-Yadkori, and C. Szepesvari. Improved regret bound and experience replay in regularized policy iteration. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6032–6042. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/lazic21a.html`.

H. Li, N. Clavette, and H. He. An analytical update rule for general policy optimization. *CoRR*, abs/2112.02045, 2021. URL `https://arxiv.org/abs/2112.02045`.

J. Luketina, S. Flennerhag, Y. Schroecker, D. Abel, T. Zahavy, and S. Singh. Meta-gradients in non-stationary environments, 2022.

J. Martens. New perspectives on the natural gradient method. *CoRR*, abs/1412.1193, 2014. URL `http://arxiv.org/abs/1412.1193`.

J. Mei, C. Xiao, B. Dai, L. Li, C. Szepesvári, and D. Schuurmans. Escaping the gravitational pull of softmax. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020a. URL `https://proceedings.neurips.cc/paper/2020/hash/f1cf2a082126bf02de0b307778ce73a7-Abstract.html`.

J. Mei, C. Xiao, C. Szepesvári, and D. Schuurmans. On the global convergence rates of softmax policy gradient methods. *CoRR*, abs/2005.06392, 2020b. URL `https://arxiv.org/abs/2005.06392`.

J. Mei, B. Dai, C. Xiao, C. Szepesvári, and D. Schuurmans. Understanding the effect of stochasticity in policy optimization. *CoRR*, abs/2110.15572, 2021a. URL `https://arxiv.org/abs/2110.15572`.

J. Mei, Y. Gao, B. Dai, C. Szepesvári, and D. Schuurmans. Leveraging non-uniformity in first-order non-convex optimization. *CoRR*, abs/2105.06072, 2021b. URL `https://arxiv.org/abs/2105.06072`.

J. Mei, W. Chung, V. Thomas, B. Dai, C. Szepesvari, and D. Schuurmans. The role of baselines in policy gradient optimization, 2023.

M. Mohri and S. Yang. Accelerating optimization via adaptive prediction, 2015.

M. Mohri and S. Yang. Accelerating online convex optimization via adaptive prediction. In A. Gretton and C. C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, volume 51 of *JMLR Workshop and Conference Proceedings*, pages 848–856. JMLR.org, 2016. URL `http://proceedings.mlr.press/v51/mohri16.html`.

T. Moskovitz, B. O'Donoghue, V. Veeriah, S. Flennerhag, S. Singh, and T. Zahavy. Reload: Reinforcement learning with optimistic ascent-descent for last-iterate convergence in constrained mdps, 2023.

A. Nemirovski. Prox-method with rate of convergence o(1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM J. Optim.*, 15(1):229–251, 2004. doi: 10.1137/S1052623403425629. URL `https://doi.org/10.1137/S1052623403425629`.

Y. Nesterov. A method for solving the convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.

J. Nocedal and S. Wright. *Numerical optimization*, pages 1–664. Springer Series in Operations Research and Financial Engineering. Springer Nature, 2006.

B. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(64)90137-5. URL `https://www.sciencedirect.com/science/article/pii/0041555364901375`.

M. L. Puterman. *Markov Decision Processes*. Wiley, 1994. ISBN 978-0471727828. URL `http://books.google.com/books/about/Markov_decision_processes.html?id=Y-gmAQAAIAAJ`.

A. Rakhlin and K. Sridharan. Online learning with predictable sequences. In S. Shalev-Shwartz and I. Steinwart, editors, *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*, volume 30 of *JMLR Workshop and Conference Proceedings*, pages 993–1019. JMLR.org, 2013a. URL `http://proceedings.mlr.press/v30/Rakhlin13.html`.

A. Rakhlin and K. Sridharan. Optimization, learning, and games with predictable sequences. *CoRR*, abs/1311.1869, 2013b. URL `http://arxiv.org/abs/1311.1869`.

A. Rakhlin and K. Sridharan. Online learning with predictable sequences, 2014.

G. Raskutti and S. Mukherjee. The information geometry of mirror descent, 2014.

B. Scherrer. Improved and generalized upper bounds on the complexity of policy iteration, 2016.

B. Scherrer, M. Ghavamzadeh, V. Gabillon, B. Lesner, and M. Geist. Approximate modified policy iteration and its application to the game of tetris. *Journal of Machine Learning Research*, 16(49):1629–1676, 2015. URL `http://jmlr.org/papers/v16/scherrer15a.html`.

J. Schmidhuber. Evolutionary principles in self-referential learning. on learning now to learn: The meta-meta-meta...-hook. Diploma thesis, Technische Universitat Munchen, Germany, 14 May 1987. URL `http://www.idsia.ch/~juergen/diploma.html`.

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. P. Lillicrap, and D. Silver. Mastering atari, go, chess and shogi by planning with a learned model. *CoRR*, abs/1911.08265, 2019. URL `http://arxiv.org/abs/1911.08265`.

J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. Trust region policy optimization. *CoRR*, abs/1502.05477, 2015. URL `http://arxiv.org/abs/1502.05477`.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL `http://arxiv.org/abs/1707.06347`.

L. Shani, Y. Efroni, and S. Mannor. Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps, 2019. URL `https://arxiv.org/abs/1909.02769`.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016a. doi: 10.1038/nature16961. URL `https://doi.org/10.1038/nature16961`.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nat.*, 529(7587):484–489, 2016b. doi: 10.1038/nature16961. URL `https://doi.org/10.1038/nature16961`.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017. doi: 10.1038/nature24270. URL `https://doi.org/10.1038/nature24270`.

E. Smirnova and E. Dohmatob. On the convergence of smooth regularized approximate value iteration schemes. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6540–6550. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/483101a6bc4e6c46a86222eb65fbcb6a-Paper.pdf`.

F. Sung, L. Zhang, T. Xiang, T. M. Hospedales, and Y. Yang. Learning to learn: Meta-critic networks for sample efficient learning. *CoRR*, abs/1706.09529, 2017. URL `http://arxiv.org/abs/1706.09529`.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, page 1057–1063, Cambridge, MA, USA, 1999. MIT Press.

M. Tomar, L. Shani, Y. Efroni, and M. Ghavamzadeh. Mirror descent policy optimization. *CoRR*, abs/2005.09814, 2020. URL `https://arxiv.org/abs/2005.09814`.

J. N. Tsitsiklis. On the convergence of optimistic policy iteration. *J. Mach. Learn. Res.*, 3:59–72, 2002. URL `http://jmlr.org/papers/v3/tsitsiklis02a.html`.

S. Vaswani, O. Bachem, S. Totaro, R. Mueller, M. Geist, M. C. Machado, P. S. Castro, and N. L. Roux. A functional mirror ascent view of policy gradient methods with function approximation. *CoRR*, abs/2108.05828, 2021. URL `https://arxiv.org/abs/2108.05828`.

V. Veeriah, M. Hessel, Z. Xu, R. L. Lewis, J. Rajendran, J. Oh, H. van Hasselt, D. Silver, and S. Singh. Discovery of useful questions as auxiliary tasks. *CoRR*, abs/1909.04607, 2019. URL `http://arxiv.org/abs/1909.04607`.

N. Vieillard, B. Scherrer, O. Pietquin, and M. Geist. Momentum in reinforcement learning. *CoRR*, abs/1910.09322, 2019. URL `http://arxiv.org/abs/1910.09322`.

N. Vieillard, T. Kozuno, B. Scherrer, O. Pietquin, R. Munos, and M. Geist. Leverage the average: an analysis of regularization in RL. *CoRR*, abs/2003.14089, 2020a. URL `https://arxiv.org/abs/2003.14089`.

N. Vieillard, B. Scherrer, O. Pietquin, and M. Geist. Momentum in reinforcement learning. In S. Chiappa and R. Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2529–2538. PMLR, 26–28 Aug 2020b. URL `https://proceedings.mlr.press/v108/vieillard20a.html`.

P. Wagner. Optimistic policy iteration and natural actor-critic: A unifying view and a non-optimality result. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL `https://proceedings.neurips.cc/paper_files/paper/2013/file/2dace78f80bc92e6d7493423d729448e-Paper.pdf`.

P. Wagner. Policy oscillation is overshooting. *Neural Networks*, 52:43–61, 2014. doi: 10.1016/j.neunet.2014.01.002. URL `https://doi.org/10.1016/j.neunet.2014.01.002`.

J. Wang and J. D. Abernethy. Acceleration through optimistic no-regret dynamics. *CoRR*, abs/1807.10455, 2018. URL `http://arxiv.org/abs/1807.10455`.

J. Wang, J. D. Abernethy, and K. Y. Levy. No-regret dynamics in the fenchel game: A unified framework for algorithmic convex optimization. *CoRR*, abs/2111.11309, 2021. URL `https://arxiv.org/abs/2111.11309`.

Y. Wang, Q. Ye, and T. Liu. Beyond exponentially discounted sum: Automatic learning of return function. *CoRR*, abs/1905.11591, 2019. URL `http://arxiv.org/abs/1905.11591`.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL `https://doi.org/10.1007/BF00992696`.

L. Xiao. On the convergence rates of policy gradient methods, 2022.

Z. Xu, H. van Hasselt, and D. Silver. Meta-gradient reinforcement learning. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2402–2413, 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/2715518c875999308842e3455eda2fe3-Abstract.html`.

Z. Xu, H. van Hasselt, M. Hessel, J. Oh, S. Singh, and D. Silver. Meta-gradient reinforcement learning with an objective discovered online. *CoRR*, abs/2007.08433, 2020. URL `https://arxiv.org/abs/2007.08433`.

Y. Ye. The simplex and policy-iteration methods are strongly polynomial for the markov decision problem with a fixed discount rate. *Mathematics of Operations Research*, 36(4):593–603, 2011. ISSN 0364765X, 15265471. URL `http://www.jstor.org/stable/41412326`.

R. Yuan, S. S. Du, R. M. Gower, A. Lazaric, and L. Xiao. Linear convergence of natural policy gradient methods with log-linear policies. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=-z9hdsyUwVQ`.

T. Zahavy, Z. Xu, V. Veeriah, M. Hessel, J. Oh, H. van Hasselt, D. Silver, and S. Singh. A self-tuning actor-critic algorithm. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/f02208a057804ee16ac72ff4d3cec53b-Abstract.html`.

Z. Zheng, J. Oh, and S. Singh. On learning intrinsic rewards for policy gradient methods. *CoRR*, abs/1804.06459, 2018. URL `http://arxiv.org/abs/1804.06459`.

M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, page 928–935. AAAI Press, 2003. ISBN 1577351894.