Stackelberg Actor-Critic: Game-Theoretic Reinforcement Learning Algorithms

Liyuan Zheng, Tanner Fiez, Zane Alumbaugh, Benjamin Chasnov, Lillian J. Ratliff

June 25, 2021

Abstract

The hierarchical interaction between the actor and critic in actor-critic based reinforcement learning algorithms naturally lends itself to a game-theoretic interpretation. We adopt this viewpoint and model the actor and critic interaction as a two-player general-sum game with a leader-follower structure known as a Stackelberg game. Given this abstraction, we propose a meta-framework for Stackelberg actor-critic algorithms where the leader player follows the total derivative of its objective instead of the usual individual gradient. From a theoretical standpoint, we develop a policy gradient theorem for the refined update and provide a local convergence guarantee for the Stackelberg actor-critic algorithms to a local Stackelberg equilibrium. From an empirical standpoint, we demonstrate via simple examples that the learning dynamics we study mitigate cycling and accelerate convergence compared to the usual gradient dynamics given cost structures induced by actor-critic formulations. Finally, extensive experiments on OpenAI gym environments show that Stackelberg actor-critic algorithms always perform at least as well and often significantly outperform the standard actor-critic algorithm counterparts.

1 Introduction

The algorithmic techniques for reinforcement learning can be classified into policy-based, value-based, and actor-critic methods (Sutton and Barto, 2018). Policy-based methods directly optimize a parameterized policy to maximize the expected return, while value-based methods estimate the expected return and then infer an optimal policy from the value-function by selecting the maximizing actions. Actor-critic methods bridge policy-based and value-based methods by learning the parameterized policy (actor) and the value-function (critic) together. In particular, actor-critic methods learn a critic that approximates the expected return of the actor while concurrently learning an actor to optimize the expected return based on the critic's estimation.

In this paper, we adopt a game-theoretic perspective of actor-critic reinforcement learning algorithms. To provide some relevant background from game theory, recall that Stackelberg games are a class of games that describe interactions between a leader and a follower (Başar and Olsder, 1998). In a Stackelberg game, the leader is distinguished by the ability to act before the follower. As a result of this structure, the leader optimizes its objective accounting for the anticipated response of the follower, while the follower selects a best response to the leader's action to optimize its own objective. The interaction between the actor and critic in reinforcement learning has an intrinsic hierarchical structure reminiscent of a Stackelberg game. Indeed, the actor aims to be at an optimum knowing that the critic responds near-optimally to the selected parameters, while the critic seeks to be at an optimum given the actor parameters or vice versa between the actor and critic. This observation forms the basis of our work which contributes a novel game-theoretic modeling framework along with theoretical and empirical results.

Modeling Contributions. We explicitly cast the interaction between the actor and critic as a two-player general-sum Stackelberg game toward solving reinforcement learning problems. Notably, this perspective deviates from the majority of work on actor-critic reinforcement learning algorithms which implicitly neglect the interaction structure by independently optimizing the actor and critic objectives using individual gradient dynamics. In order to solve the game iteratively in a manner that reflects the interaction structure, we study

learning dynamics in which the player deemed the leader updates its parameters using the total derivative of its objective defined using the implicit function theorem and the player deemed the follower updates using the typical individual gradient dynamics. We refer to this gradient-based learning method as the Stackelberg gradient dynamics. The designations of leader and follower between the actor and critic can result in distinct game-theoretic outcomes and we explore both choices and explain how the proper roles depends on the respective objective functions.

Theoretical Contributions. The Stackelberg gradient dynamics were previously studied in general nonconvex games and enjoy a number of theoretical guarantees (Fiez et al., 2020). In this paper we tailor the analysis of this learning dynamic to the reinforcement learning problem. To do this, we begin by developing a policy gradient theorem for the total derivative update (Theorem 1). Then, building off of this result, we develop a meta-framework of Stackelberg actor-critic algorithms. Specifically, this framework adapts the standard actor-critic, deep deterministic policy gradient, and soft-actor critic algorithms to be optimized using the Stackelberg gradient dynamics in place of the usual individual gradient dynamics. For the class of Stackelberg actor-critic algorithms this meta-framework admits, we prove a local convergence guarantee (Theorem 2) to a local Stackelberg equilibrium defined by gradient-based sufficient conditions.

Experimental Contributions. From an empirical standpoint, we begin by pointing out in Section 3 that the objective functions in actor-critic algorithms commonly exhibit a type of hidden structure in terms of the parameters. Given this observation, we develop simple, yet illustrative examples comparing the behavior of Stackelberg actor-critic algorithms with standard actor-critic algorithms. In particular, we observe that the Stackelberg dynamics mitigate cycling in the parameter space and accelerate convergence. We discover from extensive experiments on OpenAI gym environments that similar observations carry over to complex problems and that our Stackelberg actor-critic algorithms always perform at least as well and often significantly outperform the standard actor-critic algorithm counterparts.

2 Related Work

Game-theoretic frameworks have been studied extensively in reinforcement learning but mostly in multi-agent setting (Yang and Wang, 2020). In multi-agent reinforcement learning, the decentralized learning scheme is mostly adopted in practice (Zhang et al., 2019), where agents typically behave independently and optimize their own objective with no explicit information exchange. A shortcoming of this method is that agents fail to consider the learning process of other agents and simply treat them as a static component of the environment (Hernandez-Leal et al., 2017). To resolve this, several works design learning algorithms that explicitly account for the learning behavior of other agents (Zhang and Lesser, 2010; Foerster et al., 2018; Letcher et al., 2018), which is shown to improve learning stability and induce cooperation. In contrast, Prajapat et al. (2020) study a competitive policy optimization method for multi-agent reinforcement learning which performs recursive reasoning about the behavior of opponents to exploit them in two-player zero-sum games.

The past research taking a game-theoretic viewpoint of single-agent reinforcement learning is limited despite the fact that there is often implicitly multiple players in reinforcement learning algorithms. Rajeswaran et al. (2020) propose a framework that casts model-based reinforcement learning as a two-player general-sum Stackelberg game between a policy player and a model player. However, they only consider optimizing the objective of each player using the typical individual gradient dynamics with timescale separation as an approximation to Stackelberg gradient dynamics. Concurrent to this work, Hong et al. (2020) analyze the Stackelberg gradient dynamics with timescale separation for bilevel optimization with application to reinforcement learning. For reinforcement learning, they give a convergence guarantee for an actor-critic algorithm under assumptions such as exact linear function approximation which result in the total derivative being equivalent to the individual gradient. We provide a complimentary study by developing a general framework for Stackelberg actor-critic algorithms that we analyze without such assumptions and also extensively evaluate empirically on reinforcement learning tasks.

Single-agent reinforcement learning algorithms with second-order information trace back to natural policy gradient methods (Kakade, 2001) and the natural actor-critic algorithm (Peters and Schaal, 2008; Bhatnagar

et al., 2009). Since then, such techniques have been proposed for both policy-based and actor-critic methods (Schulman et al., 2015a, 2017; Shen et al., 2019; Tangkaratt et al., 2018). However, the gradient information in the past works do not account for the interaction between the actor and critic as in this work.

3 Motivation & Preliminaries

In this section, we begin by presenting background on Stackelberg games and the relevant equilibrium concept. Then, to motivate and illustrate the utility of Stackelberg-based actor-critic algorithms, we highlight a key hidden structure that exists in actor-critic objective formulations and explore the behavior of Stackelberg gradient dynamics in comparison to individual gradient dynamics given this design. Finally, we provide the necessary mathematical background and formalism for actor-critic reinforcement learning algorithms.

3.1 Game-Theoretic Preliminaries

A Stackelberg game is a game between two agents where one agent is deemed the leader and the other the follower. Each agent has an objective they want to optimize that depends on not only their own actions but also on the actions of the other agent. Specifically, the leader optimizes its objective under the assumption that the follower will play a best response. Let $f_1(x_1, x_2)$ and $f_2(x_1, x_2)$ be the objective functions that the leader and follower want to minimize, respectively, where $x_1 \in X_1 \subseteq \mathsf{R}^{d_1}$ and $x_2 \in X_2 \subseteq \mathsf{R}^{d_2}$ are their decision variables or strategies and $x = (x_1, x_2) \in X_1 \times X_2$ is their joint strategy. The leader and follower aim to solve the following problems:

$$\min_{x_1 \in X_1} \{ f_1(x_1, x_2) \big|\ x_2 \in \arg\min_{y \in X_2} f_2(x_1, y) \}, \tag{L}$$

$$\min_{x_2 \in X_2} f_2(x_1, x_2). \tag{F}$$

Since the leader assumes the follower chooses a best response $x_2^*(x_1) = \arg\min_y f_2(x_1, y)$,[1] the follower's decision variables are implicitly a function of the leader's. In deriving sufficient conditions for the optimization problem in (L), the leader utilizes this information by the total derivative of its cost function which is given by

$$\nabla f_1(x_1, x_2^*(x_1)) = \nabla_1 f_1(x) + (\nabla x_2^*(x_1))^\top \nabla_2 f_1(x).$$

where $\nabla x_2^*(x_1) = -(\nabla_2^2 f_2(x))^{-1} \nabla_{21} f_2(x)$. [2]

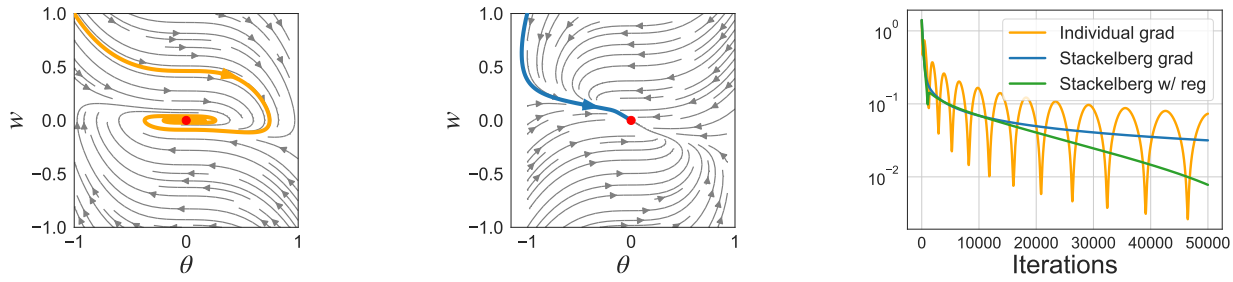
Hence, a point $x = (x_1, x_2)$ is a local solution to (L) if $\nabla f_1(x_1, x_2^*(x_1)) = 0$ and $\nabla^2 f_1(x_1, x_2^*(x_1)) > 0$. For the follower's problem, sufficient conditions for optimality are $\nabla_2 f_2(x_1, x_2) = 0$ and $\nabla_2^2 f_2(x_1, x_2) > 0$. This gives rise to the following equilibrium concept which characterizes sufficient conditions for a local Stackelberg equilibrium.

Definition 1 (Differential Stackelberg Equilibrium, Fiez et al. 2020)**.** *The joint strategy $x^* = (x_1^*, x_2^*) \in X_1 \times X_2$ is a di erential Stackelberg equilibrium if $\nabla f_1(x^*) = 0$, $\nabla_2 f_2(x^*) = 0$, $\nabla^2 f_1(x^*) > 0$, and $\nabla_2^2 f_2(x^*) > 0$.*

The Stackelberg learning dynamics derive from the first-order gradient-based sufficient conditions and are given by

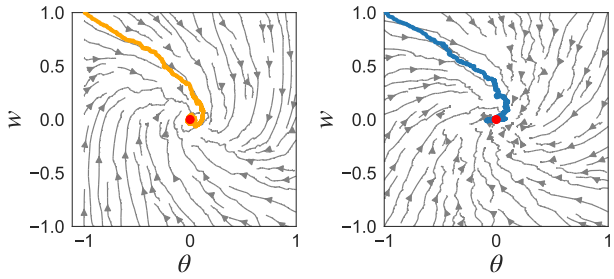
$$x_{1,k+1} = x_{1,k} - \alpha_1 \nabla f_1(x_{1,k}, x_{2,k})$$
$$x_{2,k+1} = x_{2,k} - \alpha_2 \nabla_2 f_2(x_{1,k}, x_{2,k})$$

where α_i, $i = 1, 2$ are the learning rates for the leader and follower, respectively.



(a) Individual gradient (b) Stackelberg gradient (c) Error to equilibrium

Figure 1: (a)–(b) Vector fields and trajectories of the actor and critic updates using individual gradient and Stackelberg gradient. (c) Error $\|w - w^*\|^2 + \|\theta - \theta^*\|^2$ for individual gradient, Stackelberg gradient, and Stackelberg gradient with regularization, where $(\theta^*, w^*) = (0, 0)$.



(a) Individual gradient (b) Stackelberg gradient

Figure 2: (a)–(b) Individual gradient and Stackelberg gradient with entropic regularization added to the actor objective following Soft Actor-Critic.

3.2 Motivating Examples

In the next section we present several common actor-critic formulations including the "vanilla" actor-critic, deep deterministic policy gradient, and soft actor-critic. A common theme among them is that the actor and critic objectives exhibit a simple hidden structure in the parameters. In particular, the actor objective typically has a hidden linear structure in terms of the parameters θ which is abstractly of the form $Q_w(\theta) = w^\top \mu(\theta)$. Analogously, the critic objective usually has a hidden quadratic structure in the parameters w which is abstractly of the form or $(R(\theta) - Q_w(\theta))^2$. The terminology of hidden structure in this context refers to the fact that the specified structure appears when the functions transforming the parameters are removed.[3] Interestingly, similar observations have been made regarding generative adversarial network formulations and exploited to gain insights into gradient learning dynamics for optimizing them (Vlatakis-Gkaragkounis et al., 2019; Flokas et al., 2021).

Based on this observation, we investigate simple, yet illustrative reinforcement learning problems with the aforementioned structure and compare and contrast the behavior of the Stackelberg gradient dynamics with the usual individual gradient dynamics. As we demonstrate later in Section 5, the insights we uncover from this study generally carry over to complex reinforcement learning problems.

[1]Under sufficient regularity conditions on the follower's optimization problem, the best response map is a singleton. This is a generic condition in games (Ratliff et al., 2014; Fiez et al., 2020).

[2]The partial derivative of $f(x_1, x_2)$ with respect to the x_i is denoted by $\nabla_i f(x_1, x_2)$ and the total derivative of $f(x_1, h(x_1))$ for some function h, is denoted ∇f where $\nabla f(x_1, h(x_1) = \nabla_1 f(x_1, h(x_1)) + (\nabla h(x_1))^\top \nabla_2 f(x_1, h(x_1))$.

[3]The actor and critic functions could be approximated by neural nets in practice but we consider the simplest linear case, which captures the hidden structure and gives insights for general cases.

Example. Consider a single step Markov decision process where the reward function is given by $R(\theta) = -\frac{1}{5}\theta^2$ and $\theta \in [-1, 1]$ is the decision variable of actor. Suppose that the critic is designed using the most basic linear function approximation $Q_w(\theta) = w\theta$ with $w \in [-1, 1]$. The actor seeks to find the action that maximizes the value indicated by the critic and the critic approximates the rewards of actions generated by the actor. Thus, the actor has objective $J(\theta, w) = Q_w(\theta) = w\theta$ and the critic has objective $L(\theta, w) = \mathsf{E}_{\theta\sim\rho}[(R(\theta) - Q_w(\theta))^2]$. For simplicity, we assume the critic only minimizes the mean square error of the sample action generated by current actor θ. The critic objective is then $L(\theta, w) = (R(\theta) - Q_w(\theta))^2 = (w \cdot \theta + \frac{1}{5}\theta^2)^2$.

Actor-Critic & Deep Deterministic Policy Gradient. The structure of this example closely mirrors the hidden structure of both the "vanilla" actor-critic and deep deterministic policy gradient formulations as described in the next section. The typical way to optimize the objectives is by performing individual gradient dynamics (gradient descent-ascent) on the actor and critic parameters. Figure 1(a) shows the gradient vector field and the parameter trajectories under the individual gradient dynamics. We observe that although the trajectory eventually converges to the equilibrium point $(\theta^*, w^*) = (0, 0)$, it cycles significantly. Figure 1(b) shows the vector field and parameter trajectories under the Stackelberg gradient dynamics, the details of which will be introduced in Section 4. We observe that the cycling behavior is completely eliminated as a result of the consideration given to the interaction structure. Figure 1(c) shows the error to equilibrium $\|w - w^*\|^2 + \|\theta - \theta^*\|^2$ for the individual gradient dynamics and the Stackelberg gradient dynamics along with a regularized version introduced in Section 4.5. This highlights that cycling is mitigated and convergence accelerated by optimizing using the Stackelberg gradient.

Soft Actor-Critic. The soft actor-critic algorithm also exhibits a similar structure, but with entropic regularization included in the actor objective. We show the vector fields along with the parameter trajectories for the individual gradient dynamics and the Stackelberg gradient dynamics in Figure 2(a) and Figure 2(b), respectively. Given the entropic regularization, both learning algorithms behave similarly. This perhaps indicates that the individual gradient dynamics are more well-suited to optimize this form of objectives and highlights the importance of considering how game dynamics perform on types of hidden structures when optimizing actor-critic algorithms in reinforcement learning.

Further details on the examples in this section are provided in Appendix A. Importantly, regardless of the objective function structure, the Stackelberg gradient dynamics tend to converge rather directly to the equilibrium and for some hidden structures they significantly mitigate oscillations and stabilize training. It is well-known that this is a desirable property of the reinforcement learning algorithms owing to the implications for both evaluation and real-world applications (Chan et al., 2019). Together, this motivating section suggests that introducing the Stackelberg dynamics as a "meta-algorithm" on existing actor-critic methods is likely to lead to more favorable convergence properties. We demonstrate this empirically in Section 5.

3.3 Actor-Critic Algorithms

We consider discrete-time Markov decision processes (MDPs) with continuous state space $\mathcal{S}$ and continuous action space $\mathcal{A}$. We denote the state and action at time step t by s_t and a_t, respectively. The initial state s_0 is determined by the initial state density $s_0 \sim \rho(s)$. At time step t, the agent in state s_t takes an action a_t according to a policy $a_t \sim \pi(\cdot|s_t)$ and obtains a reward $r_t = r(s_t, a_t)$. The agent then transitions to state s_{t+1} determined by the transition function $s_{t+1} \sim P(s'|s_t, a_t)$. A trajectory $\tau = (s_0, a_0, \ldots, s_T, a_T)$ gives the cumulative rewards or return defined as $R(\tau) = \sum_{t=0}^{T} \gamma^t r(s_t, a_t)$, where the discount factor $0 < \gamma \leq 1$ assigns weights to rewards received at different time steps. The expected return of π after executing a_t in state s_t can be expressed by the Q function

$$Q^\pi(s_t, a_t) = \mathsf{E}_{\tau\sim\pi}\big[\textstyle\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})|s_t, a_t\big].$$

Correspondingly, the expected return of π in state s_t can be expressed by the value function V defined as

$$V^\pi(s_t) = \mathsf{E}_{\tau\sim\pi}\big[\textstyle\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})|s_t\big].$$

The goal of reinforcement learning is to find an optimal policy that maximizes the expected return which is given by

$$\begin{aligned} J(\pi) &= \mathsf{E}_{\tau\sim\pi}\big[\textstyle\sum_{t=0}^{T}\gamma^t r(s_t,a_t)\big] = \textstyle\int_\tau p(\tau|\pi)R(\tau)\mathrm{d}\tau \\ &= \mathsf{E}_{s\sim\rho,a\sim\pi(\cdot|s)}\big[Q^\pi(s,a)\big], \end{aligned}$$

where $p(\tau|\pi) = \rho(s_0)\prod_{t=0}^{T}\pi(a_t|s_t)P(s_{t+1}|s_t,a_t)$.

The policy-based approach (Williams, 1992) parameterizes the policy π by the parameter θ and finds the optimal parameter choice θ^* by maximizing the expected return

$$J(\theta) = \mathsf{E}_{s\sim\rho,a\sim\pi_\theta(\cdot|s)}\big[Q^\pi(s,a)\big]. \tag{1}$$

This optimization problem can be solved by gradient ascent. By the policy gradient theorem (Sutton et al., 2000),

$$\nabla_\theta J(\theta) = \mathsf{E}_{s\sim\rho,a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta \log\pi_\theta(a|s)Q^\pi(s,a)\right],$$

where ∇_θ denotes the derivative with respect to θ. A common method to approximate $Q^\pi(s,a)$ in the policy gradient is by sampling trajectories and averaging returns, which is known as REINFORCE (Williams, 1992).

"Vanilla" Actor-Critic (`AC`). The actor-critic method (Konda and Tsitsiklis, 2000; Grondman et al., 2012) relies on a critic function $Q_w(s,a)$ parameterized by w to approximate $Q^\pi(s,a)$. By replacing $Q_w(s,a)$ with $Q^\pi(s,a)$ in (1), the actor which is parameterized by θ has the objective

$$J(\theta,w) = \mathsf{E}_{s\sim\rho,a\sim\pi_\theta(\cdot|s)}\big[Q_w(s,a)\big]. \tag{2}$$

The objective is optimized using gradient ascent where

$$\nabla_\theta J(\theta,w) = \mathsf{E}_{s\sim\rho,a\sim\pi_\theta(\cdot|s)}[\nabla_\theta \log\pi_\theta(a|s)Q_w(s,a)]. \tag{3}$$

The critic which is parameterized by w has the objective to minimize the mean square error between the Q-functions

$$L(\theta,w) = \mathsf{E}_{s\sim\rho,a\sim\pi_\theta(\cdot|s)}[(Q_w(s,a) - Q^\pi(s,a))^2], \tag{4}$$

where the function $Q^\pi(s,a)$ is approximated by Monte Carlo estimation or bootstrapping (Sutton and Barto, 2018).

The actor-critic method optimizes the objectives with individual gradient dynamics (Peters and Schaal, 2008; Mnih et al., 2016) which gives rise to the updates

$$\theta \leftarrow \theta + \alpha_\theta\nabla_\theta J(\theta,w), \tag{5}$$

$$w \leftarrow w - \alpha_w\nabla_w L(\theta,w), \tag{6}$$

where α_θ and α_w are the learning rates of actor and critic. Clearly, even in this basic actor-critic method, the actor and critic are coupled since J and L depend on both θ and w, which naturally lends to a game-theoretic interpretation.

Deep Deterministic Policy Gradient (`DDPG`). The `DDPG` algorithm (Lillicrap et al., 2016) is an off-policy method with subtly different objective functions for the actor and critic. In particular, the formulation has a deterministic actor $\mu_\theta(s):\mathcal{S}\to\mathcal{A}$ with the objective

$$J(\theta,w) = \mathsf{E}_{\xi\sim\mathcal{D}}\left[Q_w(s,\mu_\theta(s))\right]. \tag{7}$$

The critic objective is the mean square Bellman error

$$L(\theta,w) = \mathop{\mathsf{E}}_{\xi\sim\mathcal{D}}[(Q_w(s,a) - (r+\gamma Q_0(s',\mu_\theta(s'))))^2], \tag{8}$$

where $\xi = (s,a,r,s')$, $\mathcal{D}$ is a replay buffer, and Q_0 is a target Q network.[4]

[4]In the `DDPG` algorithm, the next-state actions used in the target network come from the target policy instead of the current policy. To be consistent with `SAC`, we use the current policy.

Algorithm 1: Stackelberg Actor-Critic Framework

Input: actor-critic algorithm ALG, player designations, and learning rate sequences $\alpha_{\theta,k}, \alpha_{w,k}$.

if actor is leader, update actor and critic in ALG with:

$$\theta_{k+1} = \theta_k + \alpha_{\theta,k} \nabla J(\theta_k, w_k) \tag{11}$$

$$w_{k+1} = w_k - \alpha_{w,k} \nabla_w L(\theta_k, w_k) \tag{12}$$

if critic is leader, update actor and critic in ALG with:

$$\theta_{k+1} = \theta_k + \alpha_{\theta,k} \nabla_\theta J(\theta_k, w_k) \tag{13}$$

$$w_{k+1} = w_k - \alpha_{w,k} \nabla L(\theta_k, w_k) \tag{14}$$

Soft Actor-Critic (SAC). The SAC algorithm (Haarnoja et al., 2018) exploits the double Q-learning trick (Van Hasselt et al., 2016) and employs entropic regularization to encourage exploration. The actor's objective $J(\theta, w)$ is

$$\mathsf{E}_{\xi\sim\mathcal{D}}\big[\min_{i=1,2} Q_{w_i}(s, a_\theta(s)) - \eta \log(\pi_\theta(a_\theta(s)|s))\big], \tag{9}$$

where $a_\theta(s)$ is a sample from $\pi_\theta(\cdot|s)$ and η is entropy regularization coefficient. The parameter of the critic is the union of both Q networks parameters $w = \{w_1, w_2\}$ and the critic objective is defined correspondingly by

$$L(\theta, w) = \mathsf{E}_{\xi\sim\mathcal{D}}\big[\textstyle\sum_{i=1,2} \left(Q_{w_i}(s,a) - y(r,s')\right)^2\big], \tag{10}$$

where

$$y(r,s') = r + \gamma(\min_{i=1,2} Q_{0,i}(s', a_\theta(s')) - \eta \log(\pi_\theta(a_\theta(s')|s'))).$$

The target networks in DDPG and SAC are updated by taking the Polyak average of the network parameters over the course of training, and the actor and critic networks are updated by individual gradient dynamics identical to (5)–(6).

4 Stackelberg Framework

In this section, we begin by formulating the actor-critic interaction as two-player general-sum Stackelberg game and introduce a Stackelberg framework for actor-critic algorithms, under which we develop novel Stackelberg versions of existing algorithms: Stackelberg actor-critic (STAC), Stackelberg deep deterministic policy gradient (STDDPG), and Stackelberg soft actor-critic (STSAC). Following this, we give a local convergence guarantee for the algorithms to a local Stackelberg equilibrium. Finally, a regularization method for practical usage of the algorithms is discussed.

4.1 Meta-Algorithm

Given an actor-critic formulation, in particular, the objectives of the actor and critic defined by $J(\theta, w)$ and $L(\theta, w)$, we can interpret the problem as a two-player general-sum Stackelberg game. If we view the actor as the leader and the critic as a follower, then the players aim to solve the following optimization problems, respectively:

$$\max_\theta \{J(\theta, w^*(\theta))\big|\ w^*(\theta) = \arg\min_{w'} L(\theta, w')\} \tag{AL}$$

$$\min_w L(\theta, w). \tag{CF}$$

On the other hand, if we view the critic as the leader and the actor as the follower, then the players aim to solve the following optimization problems, respectively:

$$\min_w \{L(\theta^*(w), w)\big|\ \theta^*(w) = \arg\max_{\theta'} J(\theta', w)\} \tag{CL}$$

$$\max_\theta J(\theta, w). \tag{AF}$$

As described in Section 3.1, we propose to optimize the objectives using a learning algorithm that accounts for the structure of the problems. Specifically, since the leader assumes the follower selects a best response, it is natural to optimize the leader objective by following the total derivative given that the follower's decision is implicitly a function of the leader's. The meta-framework we adopt for Stackelberg refinements of actor-critic methods is in Algorithm 1. The distinction compared to the usual actor-critic methods is that in the updates we replace the individual gradient for the leader by the implicitly defined total derivative which accounts for the interaction structure whereas the rest of the actor-critic method remains identical.

The dynamics with the actor as the leader are given by (11)–(12) where the actor's total derivative $J(\theta, w)$ is

$$\nabla_\theta J(\theta, w) - \nabla^\top_{w\theta} L(\theta, w)(\nabla^2_w L(\theta, w))^{-1} \nabla_w J(\theta, w). \tag{15}$$

When the critic is the leader the dynamics are given by (13)–(14) where the critic's total derivative $\nabla L(\theta, w)$ is

$$\nabla_w L(\theta, w) - \nabla^\top_{\theta w} J(\theta, w)(\nabla^2_\theta J(\theta, w))^{-1} \nabla_\theta L(\theta, w). \tag{16}$$

We now consider instantiations of this framework and explain how the total derivative can be obtained from sampling along with natural choices of leader and follower.

4.2 Stackelberg "Vanilla" Actor-Critic

We start by instantiating the Stackelberg meta-algorithm for the "vanilla" actor-critic (AC) algorithm for which the actor and critic objectives are given in (2) and (4), respectively.[5] In this on-policy formulation, the critic assists the actor in learning the optimal policy by approximating the value function of the current policy. To give an accurate approximation, the critic aims to be selecting a best response $w^*(\theta) = \arg\min_{w'} L(\theta, w')$. Thus, the actor naturally plays the role of leader and the critic the follower.

However, estimating the total derivative $\nabla J(\theta, w)$ as defined in (15) is not straightforward and we analyze each component individually. The individual gradient $\nabla_\theta J(\theta, w)$ can be computed by policy gradient theorem as given in (3). Moreover, $\nabla_w J(\theta, w) = \mathsf{E}_{s\sim\rho, a\sim\pi_\theta(\cdot|s)}[\nabla_w Q_w(s, a)]$, which follows by direct computation, and similarly

$$\begin{aligned}\nabla^2_w L(\theta, w) = \mathsf{E}_{s\sim\rho, a\sim\pi_\theta(\cdot|s)} \big[&2\nabla_w Q_w(s, a)\nabla^\top_w Q_w(s, a) \\ &+2(Q_w(s, a) - Q^\pi(s, a))\nabla^2_w Q_w(s, a)\big].\end{aligned}$$

To compute $\nabla_{w\theta} L(\theta, w)$ in (15), we begin by obtaining $\nabla_\theta L(\theta, w)$ with the following policy gradient theorem. The proof of Theorem 1 is in Appendix B.

Theorem 1. *Given an MDP and actor-critic parameters (θ, w), the gradient of $L(\theta, w)$ with respect to θ is given by*

$$\begin{aligned}\nabla_\theta L(\theta, w) = \mathsf{E}_{\tau\sim\pi_\theta}[&\nabla_\theta \log \pi_\theta(a_0|s_0) \\ &(Q_w(s_0, a_0) - Q^\pi(s_0, a_0))^2 + \textstyle\sum_{t=1}^T \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) \\ &(Q^\pi(s_0, a_0) - Q_w(s_0, a_0))Q^\pi(s_t, a_t)].\end{aligned}$$

Theorem 1 allows us to compute $\nabla_{\theta w} L(\theta, w)$ directly by $\nabla_w(\nabla_\theta L(\theta, w))$ since the distribution of $\nabla_\theta L(\theta, w)$ does not depend on w and ∇_w can be moved into the expectation.

The critic in AC is often designed to approximate the state value function $V^\pi(s)$ which has computational advantages, and the policy gradient can be computed by advantage estimation (Schulman et al., 2015b). In this formulation, $J(\theta, w) = \mathsf{E}_{\tau\sim\pi_\theta}\big[r(s_0, a_0) + V_w(s_1)\big]$ and $L(\theta, w) = \mathsf{E}_{s\sim\rho}[(V_w(s) - V^\pi(s))^2]$. Then $\nabla_\theta L(\theta, w)$ can be computed by the next proposition that is derived in Appendix C.

[5] We only demonstrate the "vanilla" actor-critic algorithm and its Stackelberg version here and in our experiments, but the framework could be generalized to more on-policy actor-critic algorithms (e.g., A2C, A3C, Mnih et al. 2016).

Proposition 1. *Given an MDP and actor-critic parameters* (θ, w)*, if the critic has the objective function* $L(\theta, w) = \mathsf{E}_{s\sim\rho}[(V_w(s) - V^\pi(s))^2]$*, then* $\nabla_\theta L(\theta, w)$ *is given by*

$$\mathop{\mathsf{E}}_{\tau\sim\pi_\theta}[2\sum_{t=0}^{T}\gamma^t\nabla_\theta \log \pi_\theta(a_t|s_t)(V^\pi(s_0) - V_w(s_0))Q^\pi(s_t, a_t)].$$

Given these derivations, terms in (15) can be estimated by sampled trajectories, and STAC updates using (11)–(12).

4.3 Stackelberg DDPG and SAC

In comparison to on-policy methods where the critic is designed to evaluate the actor using sampled trajectories generated by the current policy, in off-policy methods the critic minimizes the Bellman error using samples from a replay buffer. Thus, the leader and follower designation between the actor and critic in off-policy methods is not as clear. To this end, we propose variants of STDDPG and STSAC where the leader and follower order can be switched. Given the actor as the leader (AL), the algorithms are similar to policy-based methods, where the critic plays an approximate best response to evaluate the current actor. On the other hand, given the critic as the leader (CL), the actor plays an approximate best response to the critic value, resulting in behavior closely resembling that of the value-based methods.

As shown in (7)–(8) for DDPG and (9)–(10) for SAC, the objective functions of off-policy methods are defined in expectation over an arbitrary distribution from a replay buffer instead of the distribution induced by the current policy. Thus, each terms in the total derivatives updates in (15) and (16) can be computed directly and estimated by samples. Then, STDDPG and STSAC update using (11)–(12) or (13)–(14) depending on the choices of leader and follower.

4.4 Convergence Guarantee

Consider, without loss of generality, the actor is designated as the leader and the critic the follower. Then, the actor and critic updates with the Stackelberg gradient dynamics and learning rates sequences $\{\alpha_{\theta,k}\}, \{\alpha_{w,k}\}$ are of the form

$$\theta_{k+1} = \theta_k + \alpha_{\theta,k}(\nabla J(\theta, w) + \epsilon_{\theta,k+1}), \tag{17}$$

$$w_{k+1} = w_k - \alpha_{w,k}(\nabla_w L(\theta, w) + \epsilon_{w,k+1}), \tag{18}$$

where $\{\epsilon_{\theta,k+1}\}, \{\epsilon_{w,k+1}\}$ are stochastic processes. The results in this section assume the following.

Assumption 1. *The maps* $\nabla J : \mathsf{R}^m \to \mathsf{R}^{m_\theta}$*,* $\nabla_w L : \mathsf{R}^m \to \mathsf{R}^{m_w}$ *are Lipschitz, and* $\|\nabla J\| < \infty$*. The learning rate sequences are such that* $\alpha_{\theta,k} = o(\alpha_{w,k})$ *and* $\sum_k \alpha_{i,k} = \infty$*,* $\sum_k \alpha_{i,k}^2 < \infty$ *for* $i \in \mathcal{I} = \{\theta, w\}$*. The noise processes* $\{\epsilon_{i,k}\}$ *are zero mean, martingale di erence sequences: given the Itration* $\mathcal{F}_k = \sigma(\theta_s, w_s, \epsilon_{\theta,s}, \epsilon_{w,s},\ s \le k)$*,* $\{\epsilon_{i,k}\}_{i\in\mathcal{I}}$ *are conditionally independent,* $\mathsf{E}[\epsilon_{i,k+1}|\ \mathcal{F}_k] = 0$ *a.s., and* $\mathsf{E}[\|\epsilon_{i,k+1}\|\|\ \mathcal{F}_k] \le c_i(1 + \|(\theta_k, w_k)\|)$ *a.s. for some constants* $c_i \ge 0$ *and* $i \in \mathcal{I}$*.*

The following result gives a local convergence guarantee to a local Stackelberg equilibrium under the assumptions and the proof is in Appendix D. For this result, recall that for a continuous-time dynamical system of the form $\dot{z} = -g(z)$, a stationary point z^* of the system is said to be locally asymptotically stable or simply stable if the spectrum of the Jacobian denoted by $-Dg(z)$ is in the open left half plane.

Theorem 2. *Consider an MDP and actor-critic parameters* (θ, w)*. Given a locally asymptotically stable di erential Stackleberg equilibrium* (θ^*, w^*) *of the continuous-time limiting system* $(\dot{\theta}, \dot{w}) = (\nabla J(\theta, w), -\nabla_w L(\theta, w))$*, under Assumption 1 there exists a neighborhood* U *for which the iterates* (θ_k, w_k) *of the discrete-time system in* (17)*{*(18) *converge asymptotically almost surely to* (θ^*, w^*) *for* $(\theta_0, w_0) \in U$*.*

This result is effectively giving the guarantee that the discrete-time dynamics locally converge to a stable, game theoretically meaningful equilibrium of the continuous-time system using stochastic approximation methods given suitable learning rate sequences and unbiased gradient estimates (Borkar, 2009).

4.5 Implicit Map Regularization

The total derivative in the Stackelberg gradient dynamics requires computing the inverse of follower Hessian $\nabla_2^2 f_2(x)$. Since critic networks in practical reinforcement learning problems may be highly non-convex, $(\nabla_2^2 f_2(x))^{-1}$ can be ill-conditioned. Thus, instead of computing this term directly in the Stackelberg actor-critic algorithms, we compute a regularized variant of the form $(\nabla_2^2 f_2(x) + \lambda I)^{-1}$. This regularization method can be interpreted as the leader viewing the follower as optimizing a regularized cost $f_2(x) + \frac{\lambda}{2}\|x_2\|^2$, while the follower actually optimizes $f_2(x)$. Interestingly, the regularization parameter λ can serve to interpolate between the Stackelberg and individual gradient updates for the leader as we now formalize.

Proposition 2. *Consider a Stackelberg game where the leader updates using the regularized total derivative* $\nabla^\lambda f_1(x) = \nabla_1 f_1(x) - \nabla_{21}^\top f_2(x)(\nabla_2^2 f_2(x) + \lambda I)^{-1}\nabla_2 f_1(x)$. *As* $\lambda \to 0$ *then* $\nabla^\lambda f_1(x) \to \nabla f_1(x)$ *and when* $\lambda \to \infty$ *then* $\nabla^\lambda f_1(x) \to \nabla_1 f_1(x)$.

5 Experiments

We now show the results of extensive experiments comparing the Stackelberg actor-critic algorithms with the comparable actor-critic algorithms. We find that the actor-critic algorithms with the Stackelberg gradient dynamics always perform at least as well and often significantly outperform the standard gradient dynamics. Moreover, we provide game-theoretic interpretations of the results.

We run experiments on the OpenAI gym platform (Brockman et al., 2016) with the Mujoco Physics simulator (Todorov et al., 2012). The performance of each algorithm is evaluated by the average episode return versus the number of time steps (state transitions after taking an action according to the policy). For a fair comparison, the hyper-parameters for the actor and critic including the neural network architectures are set equal when comparing the Stackelberg actor-critic algorithms with the stand normal actor-critic algorithms. The implementation details are in Appendix E.

Performance. Figures 3(a)–3(d) show the performance of `STAC` and `AC` on several tasks. We also experiment with the common heuristic of "unrolling" the critic m steps between actor steps. For each task, `STAC` with multiple critic unrolling steps performs the best. This is due to the fact when the critic is closer to the best response, then the real response of the critic is closer to what is anticipated by the Stackelberg gradient for the actor. Interestingly, in CartPole, `STAC` with $m = 1$ performs even better than `AC` with $m = 80$.

Figures 3(e)–3(h) show the performance of `STDDPG-AL` and `STDDPG-CL` in comparison to `DDPG`. We observe that on each task, `STDDPG-AL` outperforms `DDPG` by a clear margin, whereas `STDDPG-CL` has overall better performance than `DDPG` except on `Walker2d`. Figures 3(i)–3(l) show the performance of `STSAC-AL` and `STSAC-CL` in comparison to `SAC`. For this formulation, the advantage afforded by the Stackelberg gradient is not as apparent.

In all experiments, when the actor is the leader, the Stackelberg versions either outperform or are comparable to the existing actor-critic algorithms, offering compelling evidence that the Stackelberg framework has an empirical advantage in many tasks and settings. We now provide game-theoretic interpretations of the experimental results and connect back to the examples and observations from Section 3.2.

Game-Theoretic Interpretations. `SAC` is considered the state-of-the-art model-free reinforcement learning algorithm and we observe it significantly outperforms `DDPG` (e.g., on `Hopper` and `Walker2d`). The common interpretation of its advantage is that `SAC` encourages exploration by penalizing low entropy policies. Here we provide another viewpoint.

From a game-theoretic perspective, the objective functions of `AC` and `DDPG` take on hidden linear and hidden quadratic structures for the actor and critic. This structure can result in cyclic behavior for individual gradient dynamics as shown in Section 3.2. `SAC` constructs a more well-conditioned game structure by regularizing the actor objective, which leads to the learning dynamics converging more directly to the equilibrium as seen in Section 3.2. This also explains why we observe improved performance with `STAC` and `STDDPG-AL` compared to `AC` and `DDPG`, but the performance gap between `STSAC-AL` and `SAC` is not as significant.

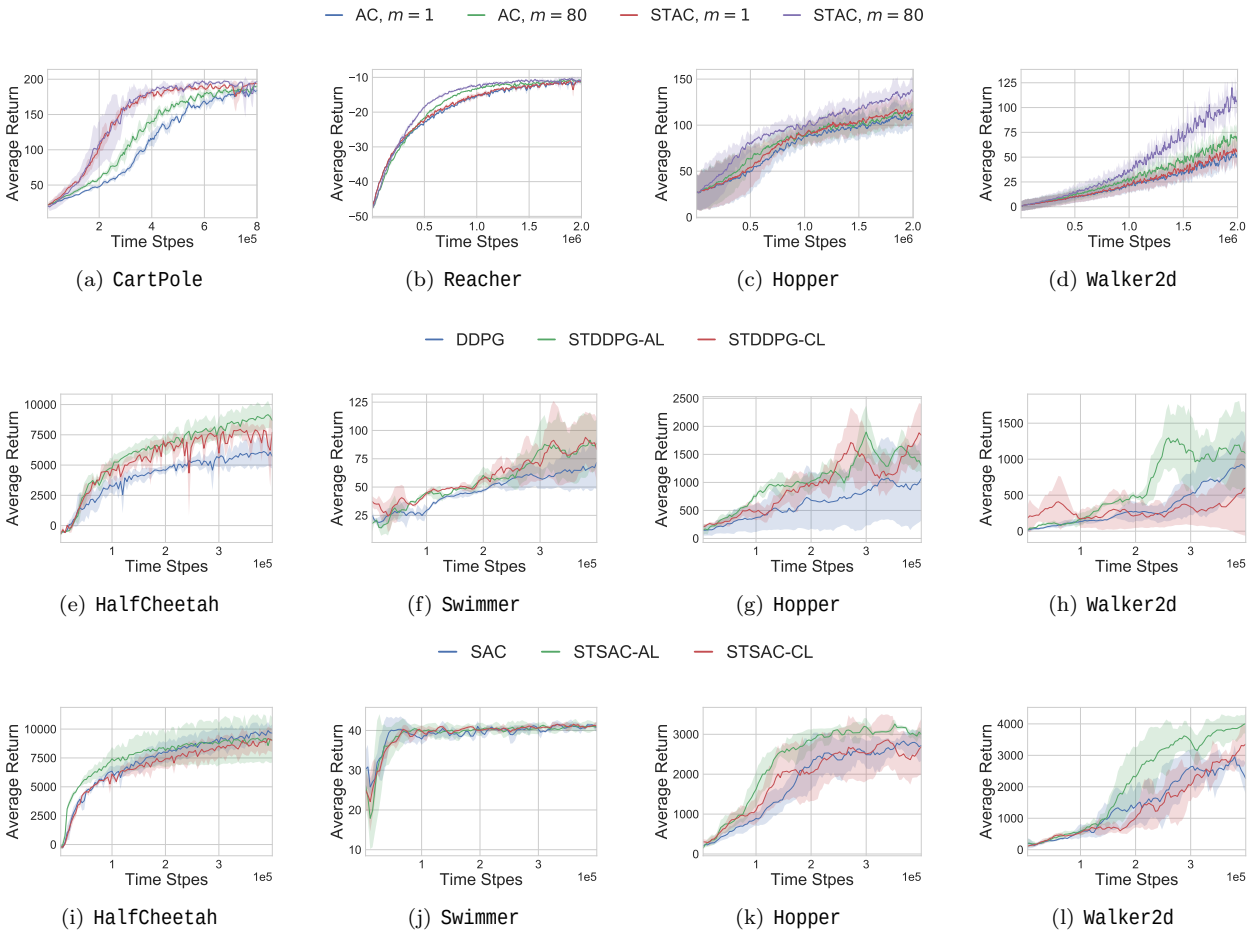


Figure 3: Comparison of AC, DDPG, SAC with their Stackelberg versions on OpenAI gym environments. Note in (a)–(d) the Stackelberg versions are red/purple and in (e)-(l) they are green/red.

Comparing AL with CL, the actor as the leader always outperforms the critic as the leader in our experiments. As described in Section 3.2, the critic objective is typically a quadratic mean square error objective which results in a hidden quadratic structure whereas the actor's objective typically is in the form of a hidden linear due to parameterization of the Q network and policy. As a result, the critic cost structure is more well-suited for computing an approximate local best response since it is more likely to be well-conditioned. Thus, the critic being the follower is a more natural hierarchical structure of the game. Unrolling the critic for multiple steps to approximate this structure and has been shown to perform well empirically (Schulman et al., 2015a). Algorithm 2 (Appendix E) shows a similar heuristic can be employed for the Stackelebrg framework.

6 Conclusion

We revisit the standard actor-critic algorithms from a game-theoretic perspective to capture the hierarchical interaction structure and introduce a Stackelberg framework for actor-critic algorithms. In this framework, we introduce novel Stackelberg versions of existing actor-critic algorithms. In experiments on a number of environments, we show that the Stackelberg actor-critic algorithms always outperform the existing counterparts when the actor plays the leader.

References

Tamer Başar and Geert Jan Olsder. *Dynamic noncooperative game theory*. SIAM, 1998.

Shalabh Bhatnagar, Richard S Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

Vivek S Borkar. *Stochastic approximation: a dynamical systems viewpoint*, volume 48. Springer, 2009.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Stephanie CY Chan, Samuel Fishman, Anoop Korattikara, John Canny, and Sergio Guadarrama. Measuring the reliability of reinforcement learning algorithms. In *International Conference on Learning Representations*, 2019.

Tanner Fiez, Benjamin Chasnov, and Lillian J Ratliff. Implicit learning dynamics in stackelberg games: Equilibria characterization, convergence analysis, and empirical study. In *International Conference on Machine Learning*, 2020.

Lampros Flokas, Emmanouil-Vasileios Vlatakis-Gkaragkounis, and Georgios Piliouras. Solving min-max optimization with hidden structure via gradient descent ascent. *arXiv preprint arXiv:2101.05248*, 2021.

Jakob Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, page 122–130, 2018.

Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307, 2012.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A survey of learning in multiagent environments: Dealing with non-stationarity. *Autonomous Agents and Multi-Agent Systems*, 2017.

Mingyi Hong, Hoi-To Wai, Zhaoran Wang, and Zhuoran Yang. A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic. *arXiv preprint arXiv:2007.05170*, 2020.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14:1531–1538, 2001.

Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

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

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representation (ICLR)*, 2016.

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

Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

Manish Prajapat, Kamyar Azizzadenesheli, Alexander Liniger, Yisong Yue, and Anima Anandkumar. Competitive policy optimization. *arXiv preprint arXiv:2006.10611*, 2020.

Aravind Rajeswaran, Igor Mordatch, and Vikash Kumar. A game theoretic framework for model based reinforcement learning. In *International conference on machine learning*, 2020.

Giorgia Ramponi and Marcello Restelli. Newton-based policy optimization for games. *arXiv preprint arXiv:2007.07804*, 2020.

Lillian J Ratliff, Samuel A Burden, and S Shankar Sastry. Genericity and structural stability of non-degenerate differential nash equilibria. In *American Control Conference*, pages 3990–3995. IEEE, 2014.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015a.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zebang Shen, Alejandro Ribeiro, Hamed Hassani, Hui Qian, and Chao Mi. Hessian aided policy gradient. In *International Conference on Machine Learning*, pages 5729–5738, 2019.

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

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Voot Tangkaratt, Abbas Abdolmaleki, and Masashi Sugiyama. Guide actor-critic for continuous control. In *International Conference on Learning Representations*, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI Conference on Arti cial Intelligence*, volume 30, 2016.

Emmanouil-Vasileios Vlatakis-Gkaragkounis, Lampros Flokas, and Georgios Piliouras. Poincaré recurrence, cycles and spurious equilibria in gradient-descent-ascent for non-convex non-concave zero-sum games. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, 2019.

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

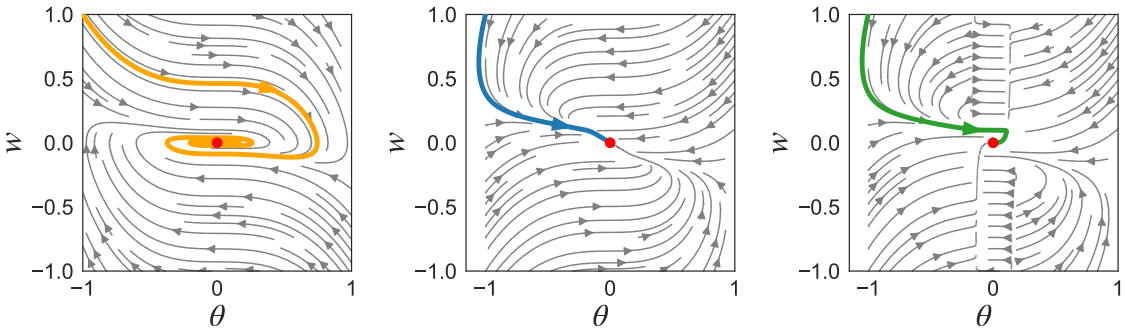
Yaodong Yang and Jun Wang. An overview of multi-agent reinforcement learning from game theoretical perspective. *Studies in Systems, Decision and Control Handbook on RL and Control*, 2020.

Chongjie Zhang and Victor Lesser. Multi-agent learning with policy prediction. In *Proceedings of the AAAI Conference on Arti cial Intelligence*, volume 24, 2010.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*, 2019.

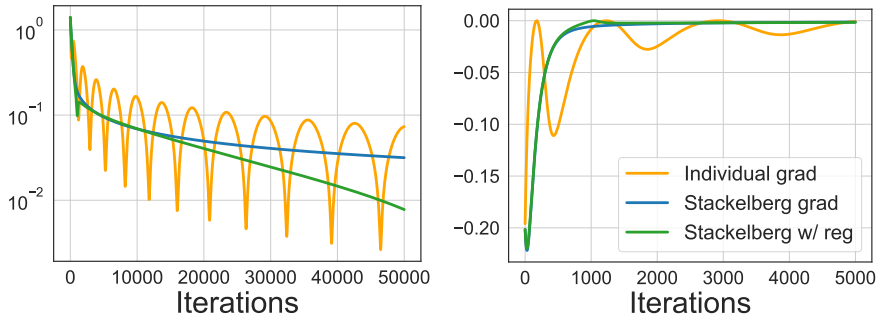
A Motivation Example Details

In this appendix section, we provide more detail for the example in Section 3.



(a) Individual gradient (b) Stackelberg gradient (c) Regularized Stackelberg gradient

Figure 4: Vector fields and trajectories of the individual gradient, Stackelberg gradient and regularized Stackelberg gradient updates. The Stackelberg updates eliminate cycling by changing the shape of the vector field.



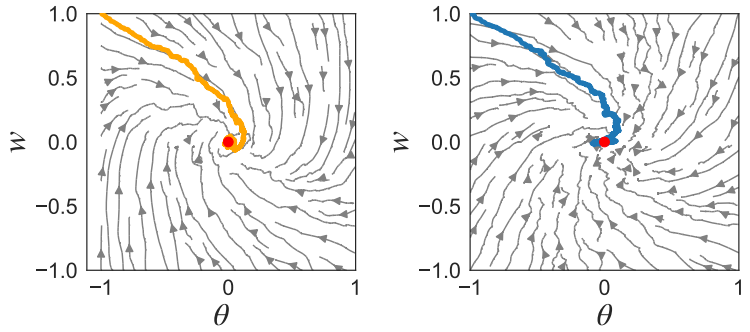
(a) Error (b) Return

Figure 5: (a) Convergence error $\|w - w^*\|^2 + \|\theta - \theta^*\|^2$ where $(\theta^*, w^*) = (0, 0)$ is the equilibrium. (b) The return $R(\theta)$ of the actor. The Stackelberg update eliminates cycling and hence, converges more directly to the equilibrium as can be seen in (a), whereas the individual gradient update oscillates significantly. Regularization helps to speed up convergence.

Recall the motivating example in which the actor plays the leader with the objective function $J(\theta, w) = w \cdot \theta$, and the critic plays the follower with objective function $L(\theta, w) = (w \cdot \theta + \frac{1}{5}\theta^2)^2$. Figure 4 shows the vector fields and trajectories of each of the updates: individual gradient play[6], Stackelberg gradient play, and regularized Stackelberg gradient play. In Figure 4(a), we observe clear cycling behavior. Such cycling behavior may be an indication of reduced reliability along the learning path and is often exacerbated by noise. Generally speaking, it is more desirable to observe smooth, monotonic changes in performance as compared to cycling behavior or noisy fluctuations around a observable trend. The reason for this is that when we go to deploy such algorithms in the real world, it can be extremely costly to have the algorithm perform in oscillatory or even unpredictable ways. This is in particular true when, as is often the case, there are unmodeled exogenous inputs or environmental factors.

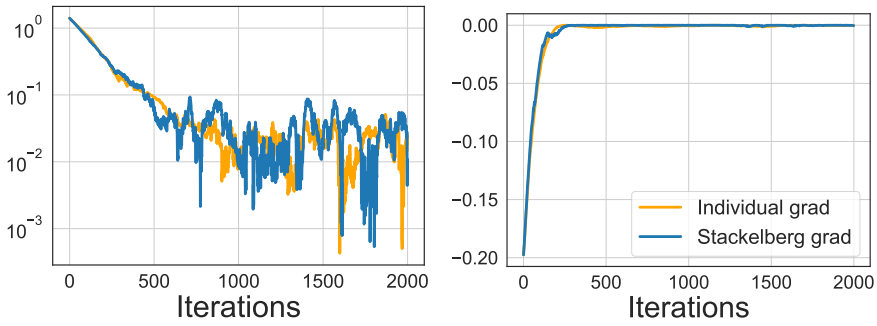
[6]In the learning in games literature, this is also often referred to as simultaneous gradient play or simultaneous gradient descent-ascent.

On the other hand, Stackelberg gradient converges more directly to the equilibrium point $(\theta^*, w^*) = (0, 0)$ and shown in both Figures 4(b) and 4(c) where the latter are the trajectories of the regularized Stackelberg gradient introduced in Section 4.5. Figure 5(a) shows the error $\|w - w^*\|^2 + \|\theta - \theta^*\|^2$ and Figure 5(b) shows the return $R(\theta)$ of each of the updates. We can observe that the cycling is mitigated and convergence accelerated by optimizing using the Stackelberg gradient, which leads to more stable returns along the learning.



(a) SAC (b) STSAC-AL

Figure 6: Vector fields and trajectories of the SAC and STSAC-AL updates.



(a) Error of SAC and STSAC-AL (b) Return of SAC and STSAC-AL

Figure 7: (a) Error for each algorithm, SAC and STSAC-AL, $\|w - w^*\|^2 + \|\theta - \theta^*\|^2$ where $(\theta^*, w^*) = (0, 0)$ is the equilibrium. (b) Return of the actor $R(\theta)$.

In Figures 6 and 7, we show the result of adding entropy regularization to the actor's objective using the SAC algorithm. Since SAC involves sampling from an stochastic policy, we plot the empirical mean gradient vector fields in Figure 6(a) and Figure 6(b), where the gradients for update are estimated by samples. With the entropy regularization, both gradient updates converge much faster and the gap between them are less significant (Figure 7(a) and 7(b)).

B Proof of Theorem 1

Recall that the critic's objective is given by $L(\theta, w) = \mathsf{E}_{s\sim\rho, a\sim\pi_\theta(\cdot|s)}[(Q_w(s,a) - Q^\pi(s,a))^2]$. The derivative is computed as follows:

$$\begin{aligned}
\nabla_\theta L(\theta, w) &= \nabla_\theta \int_{s_0} \rho(s_0) \int_{a_0} \pi_\theta(a_0|s_0) \left(Q_w(s_0,a_0) - Q^\pi(s_0,a_0)\right)^2 \mathrm{d}a_0 \mathrm{d}s_0 \\
&= \int_{s_0} \rho(s_0) \int_{a_0} \nabla_\theta \pi_\theta(a_0|s_0) \left(Q_w(s_0,a_0) - Q^\pi(s_0,a_0)\right)^2 \mathrm{d}a_0 \mathrm{d}s_0 \\
&\quad + \int_{s_0} \rho(s_0) \int_{a_0} \pi_\theta(a_0|s_0) \nabla_\theta \left(Q_w(s_0,a_0) - Q^\pi(s_0,a_0)\right)^2 \mathrm{d}a_0 \mathrm{d}s_0 \\
&= \int_{s_0} \rho(s_0) \int_{a_0} \pi_\theta(a_0|s_0) \nabla_\theta \log \pi_\theta(a_0|s_0) \left(Q_w(s_0,a_0) - Q^\pi(s_0,a_0)\right)^2 \mathrm{d}a_0 \mathrm{d}s_0 \\
&\quad + 2 \int_{s_0} \rho(s_0) \int_{a_0} \pi_\theta(a_0|s_0) \left(Q^\pi(s_0,a_0) - Q_w(s_0,a_0)\right) \nabla_\theta Q^\pi(s_0,a_0) \mathrm{d}a_0 \mathrm{d}s_0.
\end{aligned}$$

From here, it remains to compute $\nabla_\theta Q^\pi(s_0, a_0)$. To do so, recall that $Q^\pi(s_t, a_t)$ and $V^\pi(s_t)$ are given by

$$Q^\pi(s_t,a_t) = \mathsf{E}_{\tau\sim\pi}\big[\textstyle\sum_{t'=t}^T \gamma^{t'-t} r(s_{t'}, a_{t'})|s_t, a_t\big] = r(s_t,a_t) + \gamma \int_{s'} P(s'|s_t,a_t) V^\pi(s') \mathrm{d}s',$$

and

$$V^\pi(s_t) = \mathsf{E}_{\tau\sim\pi}\big[\textstyle\sum_{t'=t}^T \gamma^{t'-t} r(s_{t'}, a_{t'})|s_t\big] = \int_a \pi_\theta(a|s_t) Q^\pi(s_t,a) \mathrm{d}a.$$

Hence, $\nabla_\theta Q^\pi(s_0, a_0)$ is computed as follows:

$$\begin{aligned}
&\nabla_\theta Q^\pi(s_0,a_0) = \gamma \int_{s_1} P(s_1|s_0,a_0) \nabla_\theta V^\pi(s_1) \mathrm{d}s_1 \\
&= \gamma \int_{s_1} P(s_1|s_0,a_0) \int_{a_1} \left(\nabla_\theta \pi_\theta(a_1|s_1) Q^\pi(s_1,a_1) + \pi_\theta(a_1|s_1) \nabla_\theta Q^\pi(s_1,a_1)\right) \mathrm{d}a_1 \mathrm{d}s_1 \\
&= \gamma \int_{s_1} P(s_1|s_0,a_0) \int_{a_1} \pi_\theta(a_1|s_1) \nabla_\theta \log \pi_\theta(a_1|s_1) Q^\pi(s_1,a_1) \mathrm{d}a_1 \mathrm{d}s_1 \\
&\quad + \gamma^2 \int_{s_1} P(s_1|s_0,a_0) \int_{a_1} \pi_\theta(a_1|s_1) \int_{s_2} P(s_2|s_1,a_1) \nabla_\theta V^\pi(s_2) \mathrm{d}s_2 \mathrm{d}a_1 \mathrm{d}s_1 \\
&= \gamma \int_{s_1} P(s_1|s_0,a_0) \int_{a_1} \pi_\theta(a_1|s_1) \nabla_\theta \log \pi_\theta(a_1|s_1) Q^\pi(s_1,a_1) \mathrm{d}a_1 \mathrm{d}s_1 \\
&\quad + \gamma^2 \int_{s_1} P(s_1|s_0,a_0) \int_{a_1} \pi_\theta(a_1|s_1) \int_{s_2} P(s_2|s_1,a_1) \int_{a_2} \pi_\theta(a_2|s_2) \nabla_\theta \log \pi_\theta(a_2|s_2) Q^\pi(s_2,a_2) \mathrm{d}a_2 \mathrm{d}s_2 \mathrm{d}a_1 \mathrm{d}s_1 \\
&\quad + \gamma^3 \int_{s_1} P(s_1|s_0,a_0) \int_{a_1} \pi_\theta(a_1|s_1) \int_{s_2} P(s_2|s_1,a_1) \int_{a_2} \pi_\theta(a_2|s_2) \int_{s_3} P(s_3|s_2,a_2) \nabla_\theta V^\pi(s_3) \mathrm{d}s_3 \mathrm{d}a_2 \mathrm{d}s_2 \mathrm{d}a_1 \mathrm{d}s_1 \\
&= \gamma \int_\tau p(\tau_{1:1}|\theta) \nabla_\theta \log \pi_\theta(a_1|s_1) Q^\pi(s_1,a_1) \mathrm{d}\tau_{1:1} \\
&\quad + \gamma^2 \int_\tau p(\tau_{1:2}|\theta) \nabla_\theta \log \pi_\theta(a_2|s_2) Q^\pi(s_2,a_2) \mathrm{d}\tau_{1:2} \\
&\quad + \dots \\
&= \int_\tau \sum_{t=1}^T \gamma^t p(\tau_{1:t}|\theta) \nabla_\theta \log \pi_\theta(a_t|s_t) Q^\pi(s_t,a_t) \mathrm{d}\tau. \qquad (19)
\end{aligned}$$

where the last equality is obtained by unrolling and marginalization for the entire length of the trajectory.

Thus, coming back to the computation of $\nabla_\theta L(\theta, w)$, we have that

$$\begin{aligned}
\nabla_\theta L(\theta, w) &= \int_{s_0} \rho(s_0) \int_{a_0} \pi_\theta(a_0|s_0) \nabla_\theta \log \pi_\theta(a_0|s_0) \left(Q_w(s_0, a_0) - Q^\pi(s_0, a_0)\right)^2 \mathrm{d}a_0 \mathrm{d}s_0 \\
&\quad + 2 \int_{s_0} \rho(s_0) \int_{a_0} \pi_\theta(a_0|s_0) \left(Q^\pi(s_0, a_0) - Q_w(s_0, a_0)\right) \nabla_\theta Q^\pi(s_0, a_0) \mathrm{d}a_0 \mathrm{d}s_0 \\
&= \int_\tau p(\tau_0|\theta) \nabla_\theta \log \pi_\theta(a_0|s_0) \left(Q_w(s_0, a_0) - Q^\pi(s_0, a_0)\right)^2 \\
&\quad + 2 \sum_{t=1}^{T} \gamma^t p(\tau_{0:t}|\theta) \nabla_\theta \log \pi_\theta(a_t|s_t) \left(Q^\pi(s_0, a_0) - Q_w(s_0, a_0)\right) Q^\pi(s_t, a_t) \mathrm{d}\tau \\
&= \mathsf{E}_{\tau \sim \pi_\theta} \Bigg[\nabla_\theta \log \pi_\theta(a_0|s_0) \left(Q_w(s_0, a_0) - Q^\pi(s_0, a_0)\right)^2 \\
&\qquad\qquad + \sum_{t=1}^{T} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) \left(Q^\pi(s_0, a_0) - Q_w(s_0, a_0)\right) Q^\pi(s_t, a_t) \Bigg]
\end{aligned}$$

which completes the proof.

C Proof of Proposition 1

The critic's objective is given by $L(\theta, w) = \mathsf{E}_{s \sim \rho} \left[\left(V_w(s) - V^\pi(s)\right)^2 \right]$. Hence, taking the derivative with respect to θ, we have that

$$\begin{aligned}
\nabla_\theta L(\theta, w) &= \int_{s_0} \rho(s_0) \nabla_\theta (V_w(s_0) - V^\pi(s_0))^2 \mathrm{d}s_0 \\
&= 2 \int_{s_0} \rho(s_0) (V^\pi(s_0) - V_w(s_0)) \nabla_\theta V^\pi(s_0) \mathrm{d}s_0. \qquad (20)
\end{aligned}$$

Now we compute $\nabla_\theta V^\pi(s_0)$ in (20). Use the result of (19), we have

$$\begin{aligned}
\nabla_\theta V^\pi(s_0) &= \int_{a_0} \nabla_\theta \pi_\theta(a_0|s_0) Q^\pi(s_0, a_0) + \pi_\theta(a_0|s_0) \nabla_\theta Q^\pi(s_0, a_0) \mathrm{d}a_0 \\
&= \int_\tau \pi_\theta(a_0|s_0) \left(\nabla_\theta \log \pi_\theta(a_0|s_0) Q^\pi(s_0, a_0) + \sum_{t=1}^{T} \gamma^t p(\tau_{1:t}|\theta) \nabla_\theta \log \pi_\theta(a_t|s_t) Q^\pi(s_t, a_t) \right) \mathrm{d}\tau. \qquad (21)
\end{aligned}$$

Substituting (21) into (20), we have that

$$\begin{aligned}
\nabla_\theta L(\theta, w) &= 2 \int_\tau \sum_{t=0}^{T} \gamma^t p(\tau_{0:t}|\theta) \nabla_\theta \log \pi_\theta(a_t|s_t) \left(V^\pi(s_0) - V_w(s_0)\right) Q^\pi(s_t, a_t) \mathrm{d}\tau \\
&= \mathsf{E}_{\tau \sim \pi_\theta} \left[2 \sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) \left(V^\pi(s_0) - V_w(s_0)\right) Q^\pi(s_t, a_t) \right]
\end{aligned}$$

which completes the proof.

D Proof of Theorem 2

Without loss of generality, the actor plays the role of the leader. Consider a differential Stackelberg equilibrium of the game (θ^*, w^*) which is locally asymptotically stable[7] for the continuous time dynamical system

$$\begin{bmatrix} \dot{\theta} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} \nabla J(\theta, w) \\ -\nabla_w L(\theta, w)) \end{bmatrix}$$

where the total derivative of actor in the Stackelberg gradient is given by

$$\nabla J(\theta, w) = \nabla_\theta J(\theta, w) - \nabla_{w\theta}^\top L(\theta, w)(\nabla_w^2 L(\theta, w))^{-1} \nabla_w J(\theta, w).$$

and the individual gradient for the critic is $\nabla_w L(\theta, w)$. The actor and critic employ the discrete time updates given in Algorithm 1 where the actor is the leader. Since the actor and critic have unbiased estimates of their gradients and the learning rates are chosen as stated in Section 4.4, then the result of the theorem follows from Theorem 7 in (Fiez et al., 2020). That is, from an initial point $(\theta_0, w_0) \in U$, the Stackelberg gradient dynamics converge asymptotically to $(\theta^*, w^*) \in U$ almost surely.

Indeed, the result holds by the following reasoning. Under the assumptions on the noise processes and stepsize sequences, we treat the updates in Algorithm 1 as a stochastic approximation process (θ_k, w_k). Then, we define asymptotic pseudo-trajectories—i.e., linear interpolations between iterates (θ_k, w_k) and (θ_{k+1}, w_{k+1}). Since (θ^*, w^*) is locally asymptotically stable, there exists a neighborhood of (θ^*, w^*) and a local Lyapunov function on that neighborhood. This Lyapunov function can be used to show that the continuous time flow also starting from iterates (θ_k, w_k) and the asymptotic pseudo-trajectories are contracting onto one another asymptotically, for any sequence of iterates starting at $(\theta_0, w_0) \in U$. Hence, the iterates (θ_k, w_k), in turn, converge asymptotically to (θ^*, w^*) almost surely.

Comments on designing gradient estimators. Methods such as REINFORCE (or Monte Carlo method) provide an unbiased estimator of the follower's individual gradient. Obtaining an unbiased estimate of the total derivative for the leader, on the other hand, is a bit more nuanced. This is because there are multiple gradients being multiplied by one another in the expectation. However, as a heuristic, one way to approximate it is using the expected value of each of the terms that shows up in the total derivative.

Depending on the actor-critic algorithm and objective functions, following either Theorem 1 (Proposition 1) or direct derivatives, each term in the total derivative can be computed as an expectation over a distribution of state and action (generated by current policy in `AC` and any arbitrary policy in `DDPG` and `SAC`). Take `DDPG` as an example where $J(\theta, w) = \mathsf{E}_{\xi\sim\mathcal{D}}\left[Q_w(s, \mu_\theta(s))\right]$, and $L(\theta, w) = \mathsf{E}_{\xi\sim\mathcal{D}}\left[\left(Q_w(s,a) - (r + \gamma Q_0(s', \mu_\theta(s')))\right)^2\right]$. The second term in total derivative appears to be a multiplication of several expectations:

$$\begin{aligned}
\nabla J(\theta, w) &= \nabla_\theta J(\theta, w) - \nabla_{w\theta}^\top L(\theta, w)(\nabla_w^2 L(\theta, w))^{-1} \nabla_w J(\theta, w) \\
&= \mathsf{E}_{\xi\sim\mathcal{D}}\left[\nabla_\theta Q_w(s, \mu_\theta(s))\right] - \mathsf{E}_{\xi\sim\mathcal{D}}\left[\nabla_{w\theta}\left(\left(Q_w(s,a) - (r + \gamma Q_0(s', \mu_\theta(s')))\right)^2\right)^\top\right. \\
&\quad \left.\left(\nabla_w^2\left(\left(Q_w(s,a) - (r + \gamma Q_0(s', \mu_\theta(s')))\right)^2\right)\right)^{-1} \nabla_w Q_w(s, \mu_\theta(s))\right] \\
&\approx \mathsf{E}_{\xi\sim\mathcal{D}}\left[\nabla_\theta Q_w(s, \mu_\theta(s))\right] - \mathsf{E}_{\xi\sim\mathcal{D}}\left[\nabla_{w\theta}\left(\left(Q_w(s,a) - (r + \gamma Q_0(s', \mu_\theta(s')))\right)^2\right)\right]^\top \\
&\quad \left(\mathsf{E}_{\xi\sim\mathcal{D}}\left[\nabla_w^2\left(\left(Q_w(s,a) - (r + \gamma Q_0(s', \mu_\theta(s')))\right)^2\right)\right]\right)^{-1} \mathsf{E}_{\xi\sim\mathcal{D}}\left[\nabla_w Q_w(s, \mu_\theta(s))\right].
\end{aligned}$$

For this approximation, we can obtain an unbiased estimate by resetting the simulator as described in (Sutton et al., 2000, Chapter 11) to estimate each term in the product of expectations. As a result, this is a reasonable heuristic in practice for an approximation to the total derivative. Our policy gradient theorems also provide

[7] That is, the local linearization of the above dynamics around the point (θ^*, w^*) are in the open left-half complex plane.

Algorithm 2: Stackelberg Actor-Critic Framework with Unrolling Follower Update and Regularization

Input: actor-critic algorithm `ALG`, player designations, follower unrolling steps m, regularization hyperparameter λ, and learning rate sequences $\alpha_{1,k}, \alpha_{2,k}$.

for $k = 0, 1, 2 \dots$ **do**

  if actor is leader, **then** update actor and critic in `ALG` with

$$\theta_{k+1} = \theta_k + \alpha_{1,k}(\nabla_\theta J(\theta_k, w_{k,0}) - (\nabla^\top_{w\theta} L \circ (\nabla^2_w L + \lambda I)^{-1} \circ \nabla_w J)(\theta_k, w_{k,0}))$$
$$w_{k,l+1} = w_{k,l} - \alpha_{2,k} \nabla_w L(\theta_k, w_{k,l}), \quad l \in [0, m-1]$$
$$w_{k+1,0} = w_{k,m}$$

  if critic is leader, **then** update actor and critic in `ALG` with

$$w_{k+1} = w_k - \alpha_{1,k}(\nabla_w L(\theta_{k,0}, w_k) - (\nabla^\top_{\theta w} J \circ (\nabla^2_\theta J + \lambda I)^{-1} \circ \nabla_\theta L)(\theta_{k,0}, w_k))$$
$$\theta_{k,l+1} = \theta_{k,l} + \alpha_{2,k} \nabla_\theta J(\theta_{k,l}, w_k), \quad l \in [0, m-1]$$
$$\theta_{k+1,0} = \theta_{k,m}$$

end

us a way to derive the estimates of each of these individual terms. Obtaining unbiased estimates as an active area of research (see, e.g., Hong et al. 2020; Ramponi and Restelli 2020). Moreover, from both a theoretical and practical perspective, understanding how the batch size affects the estimate of follower Hessian and the total derivative remains open.

E Implementation Details

This section includes complete details about our experiments. Our implementation is developed based on public resource Spinning Up[8] and our source code is available at `https://anonymous.4open.science/r/51a8d354-203a-400a-b4ea-db012a74d0e9/`.

We follow the default neural network architecture used in Spinning Up. Particularly, the `AC` and `STAC` use networks of size (64, 32) with `tanh` units for both the policy and the value function. The `DDPG`, `STDDPG`, `SAC`, and `STSAC` use networks of size (256, 256) with `relu` units. The `AC` and `STAC` collected 4000 steps of agent-environment interaction per batch and use vanilla gradient descent optimizer and the `DDPG`, `STDDPG`, `SAC`, and `STSAC` use Adam optimizer with mini-batches of size 100 at each gradient descent step.

The policy gradient terms for `AC` and `STAC` are estimated by generalized average estimator (GAE) (Schulman et al., 2015b) and critics are updated by Monte Carlo method (Sutton and Barto, 2018). In discrete control task (`CartPole`), we set the Hessian regularization hyper-parameter $\lambda = 0$, and in continuous control tasks (others), we set the regularization hyper-parameter $\lambda = 500$.

The performances for `AC` and `STAC` are measured as the average trajectory return across the batch collected at each epoch. Performances for `DDPG`, `STDDPG`, `SAC`, and `STSAC` are measured once every $10,000$ steps by running the deterministic policy (or, in the case of `SAC`, the mean policy) without action noise for ten trajectories, and reporting the average return over those test trajectories.

In our Stackelberg framework, the learning rule for the leader involves computing an inverse-Hessian-vector product for the $\nabla^2_2 f_2(x_1, x_2)$ inverse term and Jacobian-vector product for the $\nabla_{12} f_2(x_1, x_2)$ terms. The second term can be computed directly by `autograd.grad` in `torch`. For the inverse-Hessian-vector term, we implement the conjugate gradient method using `autograd.grad` iteratively. This enable us to compute and estimate the total derivative on GPU directly and perform Stackelberg gradient update. In all experiments, the Stackelberg versions of actor-critic algorithms roughly take twice the time to train.

In Algorithm 2, we provide a more detailed version of our Stackelberg actor-critic algorithm framework when multiple follower unrolling steps and implicit map regularization are involved.

[8]Developed by Josh Achiam in 2018: `https://spinningup.openai.com/en/latest/`