Global Convergence in Reinforcement Learning

Federico Betti
CSE Semester Project II - EPFL Lausanne, Switzerland

December 29, 2022

Abstract

The validity of a gradient dominance property is often assumed for the convergence study of many reinforcement learning algorithms Masiha et al. [2022]. The latter condition is satisfied in weak form if the objective function satisfies a Fisher non-degeneracy assumptions together with a boundedness requirement on the compatible function approximation error. While Fisher non degeneracy holds for a large class of policies, such as Gaussian policies Liu et al. [2020], under a soft-max policy the Fisher information matrix becomes degenerate when the algorithm gets arbitrarily close to a greedy policy. A non-uniform version of the Polyak-Łojasiewicz has still been shown to hold along the trajectories of policy gradient methods Mei et al. [2020], but these algorithms require access to the full gradient of the objective which is often unfeasible to compute; on the other hand, the study in this direction for some much more practical stochastic first order and second order methods is still somehow limited. In this work, we show empirically the validity of a gradient dominance property under a soft-max policy along the trajectories of stochastic policy gradient methods and stochastic second order methods, in different discrete reinforcement learning environments. Using the results obtained, we gain further insight on the faster and more stable convergence of the stochastic cubic regularized Newton method (SCRN) over stochastic policy gradient methods. To go further in understanding the performance of second order methods, we make a first attempt at analyzing the natural policy gradient method and the differences with the cubic regularized Newton's method by comparing the second order information enclosed in the Fisher information and in the Hessian of the objective: in particular, we compare the methods on a simple MDP proposed in Kakade [2001]. Finally, we draw conclusions about the work in the final section, and we present possible developments of the study presented here. Implementation details and reproducibility of the obtained results are available at `https://github.com/federicobetti99/Global-Convergence-in-RL`.

1 Introduction

Reinforcement Learning Setup

Consider a discrete Markov decision process (MDP) $M = (\mathcal{S}, \mathcal{A}, \mathbb{P}, R, \rho, \gamma)$, where S is the state space and A is the action space. $\mathbb{P}(s'|s,a)$ denotes the probability of state transition from s to s' after taking action a and $R(\cdot,\cdot) : S \times A \to [-R_{\max}, R_{\max}]$ is a bounded reward function. $\rho : S \to [0,1]$ represents the initial distribution on the state space S and $\gamma \in (0,1)$ is the discount factor for future rewards. In this work, we restrict ourselves to discrete reinforcement learning environments.

At each time t, the agent executes an action $a_t \in \mathcal{A}$ given the current state $s_t \in \mathcal{S}$, where action selection is driven by a stochastic policy $\pi(\cdot|s) : a \in \mathcal{A} \to \pi(a|s)$, i.e., $a_t \sim \pi(\cdot|s_t)$. Then, given the state-action pair (s_t, a_t), the agent observes a reward $R_t = R(s_t, a_t)$. A fundamental quantity in reinforcement learning is the state-action value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defined as

$$Q^\pi(s,a) = \mathbb{E}_{a_t \sim \pi(\cdot|s), s_{t+1} \sim \mathbb{P}(\cdot|s_t,a_t)} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s, a_0 = a \right]. \tag{1}$$

Similarly, one can define the value function $V : \mathcal{S} \to \mathbb{R}$ defined as

$$V(s) = \mathbb{E}_{a \sim \pi(\cdot|s)} \left[Q^\pi(s,a) \right] \tag{2}$$

and the advantage function $A^\pi(s,a) = Q^\pi(s,a) - V(s)$.

In this work, we assume that the policy $\pi_\theta(\cdot|s)$ is parametrized for each state s by a vector $\theta_s \in \mathbb{R}^{|\mathcal{A}|}$. In particular, we assume a soft-max policy for action selection, namely actions in state s are sampled according to

$$\pi_\theta(a_i|s) = \frac{\exp\{\theta_s^T \mathbb{1}_{a_i}\}}{\sum_j \exp\{\theta_s^T \mathbb{1}_{a_j}\}}, \tag{3}$$

where $\mathbb{1}_{a_i}$ denotes a one-hot encoding vector of the i-th available action in state s, namely $\mathbb{1}_{a_i} = e_i$ where e_i is the i-th canonical vector.

Suppose that the initial state s_0 is drawn from some distribution ρ. Then, the reinforcement learning task is to find the optimal policy that maximizes the expected discounted return. Namely, if $\tau = \{s_t, a_t\}_{t\geq 0} \sim p(\tau|\pi_\theta)$ is a trajectory generated by the policy π_θ, with

$$p(\tau|\pi_\theta) := \rho(s_0) \prod_{t=0}^{\infty} \pi_\theta(a_t|s_t) P(s_{t+1}|s_t, a_t), \tag{4}$$

the goal of the agent is to maximize the expected return under π_θ, given by

$$J(\pi_\theta) = \mathbb{E}_{\tau\sim p(\cdot|\pi_\theta)} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{5}$$

In the sequel, we make the assumption that π_θ is differentiable with respect to θ. For notational convenience, we denote $J(\pi_\theta)$ by $J(\theta)$.

Statement of the problem

The goal of policy-based RL is to find $\theta^\star = \text{argmax}_\theta J(\theta)$. However, $J(\theta)$ may be non-concave and one settles instead for an ϵ-FOSP $\hat{\theta}$ with

$$\|\nabla_\theta J(\hat{\theta})\|^2 \leq \epsilon. \tag{6}$$

Note that because such a $\hat{\theta}$ may not lead to a large $J(\theta)$. Hence, in general, a better goal is to have

$$J(\theta^*) - J(\theta) \leq \epsilon + \mathcal{O}(\sqrt{\epsilon_{\text{bias}}}), \tag{7}$$

where ϵ_{bias} may for example reflect the inherent error related to the possibly limited expressive power of the policy parametrization π_θ. Standard policy gradient methods update the parameter θ of the policy by following the direction of the gradient. The gradient of the objective (5) is given by

$$\nabla J(\theta) = \mathbb{E}_{\tau\sim p(\cdot|\pi_\theta)} \left[\sum_{h=0}^{\infty} \left(\sum_{t=h}^{\infty} \gamma^t R(s_t, a_t) \right) \nabla \log \pi_\theta(a_h|s_h) \right] \tag{8}$$

with $\nabla \log \pi_\theta(a_h|s_h) = \mathbb{1}_{a_h} - \pi_\theta(\cdot|s_h)$ is the so-called *score function* under the soft-max policy (3). Note that this update rule recovers the well known paradigm of "activity minus expected activity" which often returns in reinforcement learning optimization algorithms. Assuming that one can explicitly compute (8), this gives rise to the policy gradient update function

$$\theta \leftarrow \theta + \alpha \nabla J(\theta), \tag{9}$$

where $\alpha > 0$ is a learning rate. In practice, most of the times, the gradient computation is possibly truncated to some finite length horizon and estimated by sample batches along the trajectories taken by the agent. The combination of the two operations aforementioned gives rise to the stochastic policy gradient methods (SPG): we describe the latter in more detail in the next subsection.
Before moving onto the presentation of the work, we still need to introduce some important quantities and related results. Let $d_\rho^{\pi_\theta}(\cdot)$ be the state visitation measure induced by policy π_θ and initial distribution ρ, which is defined as

$$d_\rho^{\pi_\theta}(s) = (1-\gamma)\mathbb{E}_{s_0\sim\rho} \left[\sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s|s_0, \pi_\theta) \right], \tag{10}$$

and let $\nu_\rho^{\pi_\theta}(\cdot,\cdot)$ the state-action visitation measure induced by the policy π_θ and initial distribution ρ, which is defined as

$$\nu_\rho^{\pi_\theta}(s,a) = (1-\gamma)\mathbb{E}_{s_0\sim\rho} \left[\sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s, a_t = a|s_0, \pi_\theta) \right], \tag{11}$$

Then, the compatible function approximation error is defined by

$$L_{\nu_\rho^{\pi_\theta}(s,a)}(w;\theta) = \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)} \left[(A^\pi(s,a) - (1-\gamma) w^T \nabla \log \pi_\theta(s,a))^2 \right]. \tag{12}$$

Definition 1.1. *(Fisher information matrix).*
The Fisher information matrix induced by a stochastic policy π_θ and an initial state distribution ρ is defined as

$$F_\rho(\theta) = \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[\nabla \log \pi_\theta(a|s) \nabla \log \pi_\theta(a|s)^T\right]. \tag{13}$$

Definition 1.2. *(Fisher non-degeneracy).*
A Fisher non-degeneracy condition is said to hold true if for all $\theta \in \mathbb{R}^d$, there exists some constant $\mu_J > 0$ such that the Fisher information matrix $F_\rho(\theta)$ defined in (13) induced by the policy π_θ and the initial state distribution ρ satisfies

$$F_\rho(\theta) \succcurlyeq \mu_J Id. \tag{14}$$

Definition 1.3. *(Gradient dominance property). Let $J(\theta) : \mathbb{R}^d \to \mathbb{R}$ a differentiable function in θ. Then J is said to satisfy a gradient dominance property if there exists $\alpha \geq 1$ such that for any $\theta \in \mathbb{R}^d$ there exists a constant τ_J and $\epsilon' > 0$ such that*

$$J(\theta^*) - J(\theta) \leq \tau_J \|\nabla J(\theta)\|_2^\alpha + \epsilon'. \tag{15}$$

Definition 1.4. *(Weak/strong gradient dominance property). Let $J(\theta) : \mathbb{R}^d \to \mathbb{R}$ a differentiable function in θ. Then J is said to satisfy a weak gradient dominance property if (15) holds with $\alpha = 1$, namely*

$$J(\theta^*) - J(\theta) \leq \tau_J \|\nabla J(\theta)\|_2 + \epsilon', \tag{16}$$

and J is said to satisfy a strong gradient dominance property if (15) holds with $\alpha = 2$, namely

$$J(\theta^*) - J(\theta) \leq \tau_J \|\nabla J(\theta)\|_2^2 + \epsilon'. \tag{17}$$

Assumption 1.1. *(Transferred compatible function approximation error). Let $u_* = (F_\rho(\theta))^\dagger \nabla J(\theta)$, where $A^\dagger$ denotes the Moore-Penrose inverse of a matrix A. For all $\theta \in \mathbb{R}^d$, there exists $\epsilon_{\text{bias}} > 0$ such that the transferred compatible function approximation error with $(s,a) \sim p(\cdot|\pi_\theta)$ satisfies*

$$\mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[(A^{\pi_\theta}(s,a) - (1-\gamma)u_*^T \nabla \log \pi_\theta(a|s))^2\right] \leq \epsilon_{\text{bias}}. \tag{18}$$

Lemma 1.1. *For a softmax tabular policy, Assumption 1.1 holds true with $\epsilon_{\text{bias}} = 0$.*

Proof. It is straightforward to verify that for a soft-max policy parametrization

$$A^{\pi_\theta}(s,a) - (1-\gamma)u_*^T \nabla \log \pi_\theta(a|s) = 0.$$

□

Theorem 1.1. *Let $\|\nabla \log \pi_\theta(a|s)\|_2 \leq G_1$ for some constant $G_1 > 0$. If Fisher non-degeneracy (14) holds true with lower bound $\mu_J > 0$ and Assumption 1.1 holds true with bias term ϵ_{bias}, a weak gradient dominance property holds true. In particular,*

$$J(\theta^*) - J(\theta) \leq \frac{G_1}{\mu_J}\|\nabla J(\theta)\| + \frac{1}{1-\gamma}\sqrt{\epsilon_{\text{bias}}}.$$

Proof. By the performance difference lemma Kakade and Langford [2002] we have

$$\mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta^*}(s,a)}\left[A^{\pi_\theta}(s,a)\right] = (1-\gamma)(J(\theta^*) - J(\theta)).$$

Using Assumption 1.1 and Jensen's inequality, we have

$$\begin{aligned}
\epsilon_{\text{bias}} &\geq \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[(A_{\pi_\theta}(s,a) - (1-\gamma)u_*^T \nabla \log \pi_\theta(a|s)^2\right] \\
&\geq \left(\mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[A_{\pi_\theta}(s,a) - (1-\gamma)u_*^T \nabla \log \pi_\theta(a|s)\right]\right)^2,
\end{aligned}$$

where $u_* = (F_\rho(\theta))^{-1}\nabla J(\theta)$. Thus

$$\begin{aligned}
\sqrt{\epsilon_{\text{bias}}} &\geq \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[A_{\pi_\theta}(s,a) - (1-\gamma)u_*^T \nabla \log \pi_\theta(a|s)\right] \\
&= (1-\gamma)(J(\theta^*) - J(\theta)) - (1-\gamma)\mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[u_*^T \nabla \log \pi_\theta(a|s)\right].
\end{aligned}$$

Rearranging terms gives

$$\begin{aligned} J(\theta^*) - J(\theta) &\leq \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[u_*^T \nabla \log \pi_\theta(a|s)\right] + \frac{1}{1-\gamma}\sqrt{\epsilon_{\text{bias}}} \\ &\leq \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[\|u_*^T\|\|\nabla \log \pi_\theta(a|s)\|\right] + \frac{1}{1-\gamma}\sqrt{\epsilon_{\text{bias}}} \\ &\leq G_1\|u_*\| + \frac{1}{1-\gamma}\sqrt{\epsilon_{\text{bias}}} \leq \frac{G_1}{\mu_J}\|\nabla J(\theta)\| + \frac{1}{1-\gamma}\sqrt{\epsilon_{\text{bias}}}. \end{aligned}$$

Thus, the claim follows. □

Remark 1.1. *For a soft-max policy parametrization, we have* $\|\nabla \log \pi_\theta(a|s)\| \leq 2$ *for every* $(s,a) \in \mathcal{S}\times\mathcal{A}$. *Furthermore, we already discussed in Lemma 1.1 that* $\epsilon_{\text{bias}} = 0$. *This implies that we can concentrate on the validity of an inequality of the form* $J(\theta^*) - J(\theta) \leq \tau_J \|\nabla J(\theta)\|_2^\alpha$ *for some* $\alpha \geq 1$.

The Fisher non-degeneracy assumption (14) is commonly used in the literature. The Fisher-non-degenerate setting implicitly guarantees that the agent is able to explore the state-action space under the considered policy class. Fisher non-degeneracy can be satisfied by certain Gaussian policies, where $\pi_\theta(\cdot|s) = \mathcal{N}(\mu_\theta(s), \Sigma)$ with the parametrized mean function $\mu_\theta(s)$ and the fixed covariance matrix $\Sigma \succcurlyeq 0$, provided that the Jacobian of $\mu_\theta(s)$ is full-row rank for all $\theta \in \mathbb{R}^d$. In addition, such an assumption holds more generally for every full-rank exponential family parametrization with their mean parameterized by $\mu_\theta(s)$ if $\mu_\theta(s)$ is full-row rank for all $\theta \in \mathbb{R}^d$. On the other hand, we know that for the softmax policy, the Fisher information matrix becomes degenerate as soon as the policy becomes deterministic, which in turn means that close to the optimum the constant τ_J may explode or not be well defined. Without the Fisher non-degeneracy assumption, establishing a gradient dominance property or any convergence property of the algorithms becomes difficult.

Previous work have shown that a non-uniform version of (16) still holds for the soft-max tabular policy.

Theorem 1.2. *Assume that the initial state distribution satisfies* $\min_s \rho(s) > 0$. *Furthermore, denote by* $a^*(s)$ *the optimal action in state* s *under a deterministic policy* π_{θ^*}. *Then it holds that*

$$\|\nabla J(\theta_t)\|_2 \geq \frac{\min_s \pi_{\theta_t}(a^*(s)|s)}{\sqrt{|\mathcal{S}|}\ \ \|d_\rho^{\pi_{\theta^*}}/d_\rho^{\pi_{\theta_t}}\|_\infty} \left(J(\theta^*) - J(\theta_t)\right). \tag{19}$$

Assuming $\min_s \rho(s) > 0$, we guarantee that $\|d_\rho^{\pi_{\theta^*}}/d_\rho^{\pi_{\theta_t}}\|_\infty > 0$, and hence the denominator of the PL constant is well defined. However, this gradient dominance inequality is cumbersome and suboptimal as it is not possible, in general, to bound from below the optimal action probabilities given by $\min_s \pi_\theta(a^*(s)|s)$. The difficulties in doing so, especially in the stochastic setting, were already highlighted in Agarwal et al. [2020]. For the deterministic case, when the exact PG is available, Mei et al. [2020] have shown that along the trajectories of the policy gradient algorithm for a softmax policy, $\min_s \pi_\theta(a^*(s)|s) > 0$ and a weak gradient dominance property holds true.

Lemma 1.2. *Using the update rule* (9), *it holds that* $c = \inf_{s\in\mathcal{S}, t\geq 1} \pi_{\theta_t}(a^*(s)|s) > 0$.

To penalize deterministic policies, entropy regularized reinforcement learning considers a surrogate objective

$$J^\lambda(\theta) = J(\theta) + \lambda \mathcal{H}(p, \pi), \tag{20}$$

for a regularization parameter $\lambda > 0$ which determines the strength of the penalty and $J(\theta)$ the usual objective defined in (5). In (20), $\mathcal{H}(p,\pi)$ is the discounted entropy defined as

$$\mathcal{H}(p,\pi) = \mathbb{E}_{\tau\sim p(\cdot|\pi_\theta)}\left[\sum_{t=0}^{\infty} -\gamma^t \log \pi_\theta(a_t|s_t)\right]$$

For an entropy regularized objective Grandvalet and Bengio [2006], a better version of the non-uniform Lojasiewicz-inequality above holds true.

Lemma 1.3. *Let* $J^\lambda(\theta)$ *denote the entropy regularized objective* (20), *and assume that* $\rho(s) > 0$ *for all* $s \in \mathcal{S}$. *Then it holds that*

$$\|\nabla J^\lambda(\theta_t)\|_2^2 \geq \frac{2\lambda\ \ \min_s \rho(s)\ \ \min_{s,a}(\pi_{\theta_t}(a|s))^2}{|\mathcal{S}|\ \ \|d_\rho^{\pi_{\theta_\lambda^*}}/d_\rho^{\pi_{\theta_t}}\|_\infty} \left(J^\lambda(\theta^*) - J^\lambda(\theta_t)\right), \tag{21}$$

where the index λ *for* π_θ *denotes the fact that* $\pi_{\theta_\lambda^*}$ *is the policy which optimizes the entropy regularized objective.*

Remark 1.2. *Due to the presence of regularization, the optimal solution will be biased with the bias disappearing as $\lambda \to 0$. More precisely, the optimal policy $\pi_{\theta^*_\lambda}$ of the entropy-regularized problem could also be nearly optimal in terms of the unregularized objective function, as long as the regularization parameter λ is chosen to be small. In particular, concerning the difference between the two objectives* (5) *and* (20)*, one can show that for any initial state distribution ρ it holds that*

$$J(\theta^*_\lambda) \leq J(\theta^*) \leq J(\theta^*_\lambda) + \lambda \frac{\log |\mathcal{A}|}{1-\gamma},$$

where θ^*_λ is the maximizer of the regularized objective.

Lemma 1.4. *Assume that the initial state distribution satisfies $\min_s \rho(s) > 0$. Then along the trajectories $\{\theta_t\}$ generated by the entropy regularized policy gradient update it holds that $c := \inf_{t \geq 1} \min_{s,a} \pi_\theta(a|s) > 0$.*

Remark 1.3. *The main difference to the previous versions of the non-uniform Łojasiewicz inequality is that the sub-optimality gap appears under the square root. For small sub-optimality gaps this means that the gradient must be larger, i.e. a stronger "signal".*

Having established under which conditions a gradient dominance property holds true with $\alpha = 1$ for the objective (5) and with $\alpha = 2$ for the entropy regularized objective (20), we now introduce some technical definitions to claim the convergence rates proved in Mei et al. [2020] upon the validity of (15). A commonly used assumption which is satisfied by the expected return (20) for $\lambda \neq 0$ and $\lambda = 0$ is a smoothness assumptions, which guarantees boundedness of the Hessian norm. We show that this condition holds true for the expected return under a soft-max policy. Here, for space reasons, we only show basic results on the log-policy function (note that this is sufficient) and refer the interested reader to Agarwal et al. [2021].

Definition 1.5. *(Smooth function). A function $f : \mathbb{R}^d \to \mathbb{R}$ is said to be smooth if there exists a constant $L > 0$ such that for any $x, y \in \mathbb{R}^d$ it holds that*

$$f(x) \leq f(y) + \nabla f(y)^T (x - y) + \frac{L}{2} \|x - y\|_2^2. \tag{22}$$

For the expected return (5) without any entropy regularization, it is easy to show that

$$\frac{\partial \log \pi_\theta(\cdot|s)}{\partial \theta_s} = \mathbb{1}_a - \pi_\theta(\cdot|s), \quad \frac{\partial^2 \log \pi_\theta(\cdot|s)}{\partial \theta_s^2} = \pi_\theta(\cdot|s)\pi_\theta(\cdot|s)^T - \mathrm{Diag}\left(\pi_\theta(\cdot|s)\right). \tag{23}$$

This immediately gives the following results.

Theorem 1.3. *The log-policy function is 2-smooth, namely for any $\theta, \theta' \in \mathbb{R}^d$ we have*

$$\|\nabla \log \pi_\theta(\cdot|s) - \nabla \log \pi_{\theta'}(\cdot|s)\| \leq 2\|\theta - \theta'\|_2. \tag{24}$$

Proof. We compute

$$\|\nabla \log \pi_\theta(\cdot|s) - \nabla \log \pi_{\theta'}(\cdot|s)\| = \|\mathbb{1}_a - \pi_\theta(\cdot|s) - \mathbb{1}_a + \pi_{\theta'}(\cdot|s)\| = \|\pi_\theta(\cdot|s) - \pi_{\theta'}(\cdot|s)\| \overset{(a)}{\leq} 2\|\theta - \theta'\|_2 \tag{25}$$

where at (a) we used the fact that $\pi_\theta(\cdot|s)$ is 2-Lipschitz. □

Lemma 1.5. *The objective* (5) *has smoothness constant $L = \frac{8}{(1-\gamma)^3}$, namely*

$$\|\nabla J(\theta_1) - \nabla J(\theta_2)\|_2 \leq \frac{8}{(1-\gamma)^3} \|\theta_1 - \theta_2\|_2$$

Remark 1.4. *Similar results can be proven for the entropy regularized objective which we introduced in* (20)*. In particular, the smoothness constant in the latter case is $\frac{8+4\lambda(1+2\log|\mathcal{A}|)}{(1-\gamma)^3}$, which recovers the previous result for a temperature parameter $\lambda = 0$.*

Combining theorem 1.2 and Lemma 1.2, we conclude that a weak gradient dominance holds along the trajectories followed by policy gradient methods. Upon this condition and L-smoothness of the objective, Mei et al. [2020] show that policy gradient converges to the optimal policy at a rate $\mathcal{O}\left(\frac{1}{t}\right)$ Similarly, a strong gradient dominance property holds (namely with $\alpha = 2$) for entropy regularized policy gradient methods from Theorem 1.3 and Lemma 1.4. Together with smoothness of the objective function, taking a learning rate $\alpha_t \leq \frac{1}{L}$ yields a convergence rate $\mathcal{O}\left(e^{-ct}\right)$ to the optimum. In practice, the knowledge

of the full gradient of the objective is often not available, so one uses instead sample batches from the environment to build unbiased estimates of the gradient to be used in the update of the policy parameters θ. However, to our knowledge, for stochastic methods the validity of a weak gradient dominance property has not been established nor studied exhaustively under a soft-max policy for action selection. As a consequence, in this work we are interested in studying empirically and theoretically whether along the trajectories of stochastic first order and second order methods the PL constant $\tau_J(\theta)$ can be bounded from above. Namely, we study whether for a sequence $\{\theta_t\}$ generated by a stochastic optimization algorithm it holds that for all $t \geq 1$

$$J(\theta^*) - J(\theta_t) \leq \tau_J(\theta_t)\|\nabla J(\theta_t)\|,$$

for $\lambda = 0$ in (20), or whether

$$J^\lambda(\theta^*) - J^\lambda(\theta_t) \leq \tau_{J^\lambda}(\theta_t)\|\nabla J^\lambda(\theta_t)\|^2$$

for the entropy regularized objective (20), for a positive constant $\tau_J(\theta_t) \leq C$ bounded from above by some $C > 0$.

Remark 1.5. *For a softmax policy, we already established previously that $\epsilon' = 0$ in (16). Hence, we can study directly only the boundedness of the PL constant in front of $\|\nabla J(\theta)\|^\alpha$ for some coefficient $\alpha \geq 1$.*

In particular, in this work we focus on the comparison and the validity of (15) in a weak or a strong form for three stochastic algorithms, plus a variance reduced variant: vanilla stochastic policy gradient (SPG), entropy regularized vanilla stochastic policy gradient (ESPG), two stages entropy regularized stochastic policy gradient Ding et al. [2021] and stochastic cubic regularized Newton (SCRN) Tripuraneni et al. [2018]. For an exhaustive presentation of the stochastic policy gradient method, we refer the interested reader to Yuan et al. [2022], and similarly for the entropy regularized objective function Grandvalet and Bengio [2006]. The two stages entropy regularized SPG algorithm chooses a large enough batch size in the very first episodes to escape flat regions and to mitigate for the non-coercive landscape of the objective function: indeed, the search direction may be dominated by the gradient estimation noise at the region where the landscape is highly flat. On the other hand, after a certain number of iterations, which is a constant with respect to the optimality gap, the iteration will reach a region where the landscape has enough curvature information: as a consequence, a smaller batch size at the end of training is sufficient to guarantee a fast convergence to the optimal policy. For this algorithm, a strong gradient dominance property holds with high probability. For space reasons and to focus on the main results of the work, we omit the derivation of such a result, referring the interested reader to Ding et al. [2021]. What is important for us is that, if we have a theoretical guarantee that the action probabilities are bounded away from zero under a softmax policy with high probability, comparing the validity of a gradient dominance property for the other stochastic algorithms we consider in this work with respect to how the PL constant evolves for the latter algorithm can give nice insight on the effective validity of the property along their trajectories. While we do not present a detailed presentation for the two stages algorithm, in the next subsection, starting from the use of second order methods in machine learning tasks, we introduce the SCRN algorithm and its motivation.

Stochastic Cubic Regularized Newton

In machine learning tasks, we can often recast the training goal as the minimization of an empirical risk, namely we want to solve the optimization problem

$$\min_x f(x), \quad x \in \mathbb{R}^d. \tag{26}$$

For simplicity, we assume the problem to be unconstrained, but similar reasoning apply to constrained optimization tasks. To solve (26), second order methods have been shown to outperform first order methods in a variety of machine learning tasks. However, their main bottlenecks are usually the expensive access to the second order information and the higher requirements on the objective function.

Definition 1.6. *(Strongly convex function). A function $f : \mathbb{R}^d \to \mathbb{R}$ is said to be strongly convex if there exists a constant $\mu > 0$ such that for any $x, y \in \mathbb{R}^d$ it holds*

$$f(x) \geq f(y) + \nabla f(y)^T(x - y) + \frac{\mu}{2}\|x - y\|_2^2. \tag{27}$$

Assumption 1.2. *(Lipschitz Hessians). There exists a constant $\rho > 0$ such that*

$$\|\nabla^2 f(x) - \nabla^2 f(y)\|_2 \leq \rho\|x - y\|_2,$$

for all $x, y \in \mathbb{R}^d$.

Remark 1.6. *If a twice differentiable function f is μ-strongly convex, this implies in particular that for every $x \in \mathbb{R}^d$ $\|\nabla^2 f(x)^{-1}\| \leq \frac{1}{\mu}$. Hence, the Hessian is always non-degenerate and the Newton step $x_{t+1} = x_t - (\nabla^2 f(x_t))^{-1}\nabla f(x_t)$ is always well defined.*

For a strongly convex objective with ρ-Lipschitz Hessians $f : \mathbb{R}^d \to \mathbb{R}$, Nesterov and Polyak have shown that Newton's method converges to the global minimum x^* at a quadratic rate, namely

$$f(x_{t+1}) - f(x^*) \leq \frac{\rho}{2}\|x_t - x^*\|_2^2, \tag{28}$$

where $\{x_t\}$ is the sequence generated by the method. Even when requiring less regularity on the objective, second order information boosts training. Many works, see e.g. Battiti [1992], have been trying to understand deeply the reasons behind this. Empirically, they seem to avoid saddle points more efficiently than gradient-based methods because of the curvature information which is extracted from the landscape at each iteration.

A cubic regularized version of the Newton's method was introduced by Nesterov and Polyak who showed a local convergence rate on functions satisfying a strong gradient dominance property Nesterov and Polyak [2006], namely condition (17). We now introduce the main idea behind the algorithm before moving to its application in a reinforcement learning setting. At every step, gradient descent finds the minimizer of a local second-order Taylor expansion at each step, i.e.

$$x_{t+1}^{GD} = \text{argmin}_x \left[f(x_t) + \nabla f(x_t)^T (x - x_t) + \frac{L}{2}\|x - x_t\|^2 \right]. \tag{29}$$

Note that in (29), the second order term of the objective minimized at every step is only a very loose upper bound for the curvature information embedded in the landscape of f. Differentiating the above in x gives rise to the usual update rule

$$x_{t+1}^{GD} = x_t - \frac{1}{L}\nabla f(x_t).$$

On the other hand, the cubic regularized Newton method finds the minimizer of a local third-order Taylor expansion given by

$$x_{t+1}^{CRN} = \text{argmin}_x \left[f(x_t) + \nabla f(x_t)^T (x - x_t) + \frac{1}{2}(x - x_t)^T \nabla^2 f(x_t)(x - x_t) + \frac{\rho}{6}\|x - x_t\|^3 \right], \tag{30}$$

where ρ is the Lipschitz constant for the Hessian of f. Unlike the previous case, the previous problem does not admit a closed form solution, but the optimal x is usually approached by gradient descent on the objective (30) instead. We discuss this more in detail in the stochastic setting below.

Theorem 1.4. *The log-policy function has 6-Lipschitz Hessians, i.e. for any $\theta, \theta' \in \mathbb{R}^d$ we have*

$$\|\nabla^2 \log \pi_\theta(\cdot|s) - \nabla^2 \log \pi_{\theta'}(\cdot|s)\| \leq 6\|\theta - \theta'\|_2. \tag{31}$$

Proof. We compute

$$\begin{aligned}
\|\nabla^2 \log \pi_\theta(\cdot|s) - \nabla^2 \log \pi_{\theta'}(\cdot|s)\| &= \|\pi_\theta(\cdot|s)\pi_\theta(\cdot|s)^T - \text{Diag}\left(\pi_\theta(\cdot|s)\right) - \pi_{\theta'}(\cdot|s)\pi_{\theta'}(\cdot|s)^T + \text{Diag}\left(\pi_{\theta'}(\cdot|s)\right)\| \\
&\leq \|\pi_\theta(\cdot|s)\pi_\theta(\cdot|s)^T - \pi_{\theta'}(\cdot|s)\pi_{\theta'}(\cdot|s)^T\| + \|\text{Diag}\left(\pi_\theta(\cdot|s)\right) - \text{Diag}\left(\pi_{\theta'}(\cdot|s)\right)\| \\
&\leq \|\pi_\theta(\cdot|s)\| \;\; \|\pi_\theta(\cdot|s) - \pi_{\theta'}(\cdot|s)\| + \|\pi_{\theta'}(\cdot|s)\| \;\; \|\pi_\theta(\cdot|s) - \pi_{\theta'}(\cdot|s)\| \\
&\leq 4\|\theta - \theta'\|_2 + 2\|\theta - \theta'\|_2 = 6\|\theta - \theta'\|_2
\end{aligned}$$

where to bound the second term we used Theorem 1.3. □

Remark 1.7. *Theorem 1.4 can be used to verify that the objective $J(\theta)$ has Lipschitz Hessians with constant $\rho(\gamma) = \mathcal{O}\left(\frac{1}{(1-\gamma)^3}\right)$. For space reasons and to illustrate only the main parts of the works, we omit the full derivation which can be found in Masiha et al. [2022].*

Given the smoothness assumptions verified for the objective at hand in a reinforcement learning setting in Theorem 1.3, we now pose our attention to the presentation of the cubic regularized Newton algorithm. Most previous work on the cubic-regularized Newton method has focused on the non-stochastic or partially-stochastic settings. In this work, we are interested in its stochastic version introduced in

Tripuraneni et al. [2018]. Let us present the algorithm applied directly on the expected return (5) which we consider for the task at hand. For the problem of stochastic optimization of a non-concave objective

$$\max_{\theta} J(\theta) = \mathbb{E}_{\tau \sim p(\cdot|\pi_\theta)} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right], \tag{32}$$

In practice, one cannot compute (8) but the latter quantity is commonly truncated up to a length horizon H, namely

$$\nabla J_H(\theta) = \mathbb{E} \left[\sum_{h=0}^{H-1} \left(\sum_{t=h}^{H-1} \gamma^t R(s_t, a_t) \right) \nabla \log \pi_\theta(a_h|s_h) \right]. \tag{33}$$

The truncation bias introduced in this step can be fully characterized as long as the log-policy has bounded gradients, i.e. there exists a constant $G_1 > 0$ such that $\|\nabla \log \pi_\theta\|_2 \leq G_1$. This was already verified for the soft-max policy in 1.3 with $G_1 = 2$. Upon this assumptions, Masiha et al. [2022] show that

$$\|\nabla J(\theta) - \nabla J_H(\theta)\|_2 \leq \frac{G_1}{1-\gamma} \left(\frac{1}{1-\gamma} + H \right)^{1/2} \gamma^H.$$

Assume that we sample m trajectories $\tau^i = \{s_t^i, a_t^i\}_{t \geq 0}$, $1 \leq i \leq m$, then we define an unbiased estimator of $\nabla J_H(\theta)$ as

$$g := \hat{\nabla}_m J_H(\theta) = \frac{1}{m} \sum_{i=1}^{m} \sum_{h=0}^{H-1} \left(\sum_{t=h}^{H-1} \gamma^t R(s_t^i, a_t^i) \right) \nabla \log \pi_\theta(a_h^i|s_h^i). \tag{34}$$

Stochastic cubic regularized Newton needs also an unbiased estimator of the Hessian matrix $\nabla^2 J_H(\theta)$ to be passed to the cubic subsolver. It can be shown Shen et al. [2019] that

$$\nabla^2 J_H(\theta) = \mathbb{E} \left[\nabla \phi(\theta; \tau) \nabla \log p(\tau|\pi_\theta)^T + \nabla^2 \phi(\theta; \tau) \right], \tag{35}$$

where $\phi(\theta; \tau) = \sum_{h=0}^{H-1} \left(\sum_{t=h}^{H-1} \gamma^t R(s_t, a_t) \right) \log \pi_\theta(a_h|s_h)$. Under the assumption of bounded Hessians, i.e. there exists a constant G_2 such that $\|\nabla^2 \log \pi_\theta\|_2 \leq G_2$, we have that the truncation bias satisfies

$$\|\nabla^2 J(\theta) - \nabla^2 J_H(\theta)\|_2 \leq \frac{G_2 + G_1^2}{1-\gamma} \left(\frac{1}{1-\gamma} + H \right) \gamma^H.$$

Again, for a softmax policy we have $G_2 = 1$, so the previous result applies. Thus, an unbiased estimator of $\nabla^2 J_H(\theta)$ is

$$B := \hat{\nabla}_m^2 J_H(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla \phi(\theta; \tau^i) \nabla \log p(\tau^i|\pi_\theta) + \nabla^2 \phi(\theta; \tau^i). \tag{36}$$

Furthermore, one can show that

$$\mathbb{E} \left[\|\nabla J_H(\theta) - \nabla_m J_H(\theta)\|_2^2 \right] \leq \frac{H G_1}{m(1-\gamma)^2}, \quad \mathbb{E} \left[\|\nabla^2 J_H(\theta) - \nabla_m^2 J_H(\theta)\|_2^2 \right] \leq \frac{4e(H^2 G_1^4 + G_2^2)}{m(1-\gamma)^4}.$$

From the triangular inequality, i.e.

$$\|\nabla J(\theta) - \nabla_m J_H(\theta)\|_2 \leq \|\nabla J_H(\theta) - \nabla_m J_H(\theta)\|_2 + \|\nabla J(\theta) - \nabla J_H(\theta)\|_2$$

and exploiting the previous results, one can derive an expression for the expected deviation of the estimators (34) and (36) from the true expressions $\nabla J(\theta)$ and $\nabla^2 J(\theta)$.

Now that we defined how to properly deal with the stochasticity of the the gradients and of the Hessians via the definition of the stochastic unbiased estimators (34) and (36), we can approach the solution to (30) by solving alternatively the *stochastic cubic submodel* given by

$$x_{t+1}^{SCRN} = \text{argmin}_x m(x, x_t) \tag{37}$$

where

$$m(x, x_t) = \text{argmin}_x \left[f(x_t) + g_t^T (x - x_t) + \frac{1}{2}(x - x_t)^T B_t (x - x_t) + \frac{\rho}{6} \|x - x_t\|^3 \right]. \tag{38}$$

In general, as there is no closed form solution for the solution to (30), the same holds for (37). Thus, one can employ at each iteration gradient descent to approximate the optimal x. For an exhaustive

description of the algorithm, we refer the interested reader to Tripuraneni et al. [2018]. Here, for space reasons, we restrict to presenting the pseudocode of the full algorithm in Algorithm 1. Similarly, the pseudocode for the Cubic Subsolver to approximate the solution to (37) is given in Algorithm 2. In both pseudocodes, the notation is the same used in the above, in particular L is always a Lipschitz constant for the gradient of the objective function and ρ is the Lipschitz constant for the Hessian of the objective function (5). We refer to the numerical results section for an exhaustive list of the hyper-parameters used during the training runs and the simulations.

Algorithm 1: Stochastic Cubic Regularized Newton with stopping criterion

Data: Batch sizes n_1, n_2, initial guess x_0, accuracy ϵ, cubic penalty parameter ρ, maximum number of iterations T

$t \leftarrow 1$
$\|\Delta_0\| = \infty$
while $\|\Delta_{t-1}\| \geq \sqrt{\epsilon}$ *or* $t \leq T$ **do**
  $g_t \leftarrow \frac{1}{n_1}\sum_{i=1}^{n_1} \nabla f(x_t, \xi_i)$ ▷ `Sample gradients`
  $B_t \leftarrow \frac{1}{n_2}\sum_{i=1}^{n_2} \nabla^2 f(x_t, \xi_i)$ ▷ `Sample Hessians`
  $\Delta_t = \operatorname{argmin}_{\Delta \in \mathbb{R}^d} \langle g_t, \Delta \rangle + \frac{1}{2}\langle \Delta, B_t \Delta \rangle + \frac{\rho}{6}\|\Delta\|^3$ ▷ `Cubic Subsolver`$(g_t, B_t[\cdot], \epsilon)$ 2
  $x_{t+1} = x_t + \Delta_t$ ▷ `Update` x
  $t \leftarrow t + 1$
end

Algorithm 2: Cubic Subsolver

Data: $g, B[\cdot], \epsilon$
if $\|g\| \geq \frac{L^2}{\rho}$ **then**
  $R_c \leftarrow -\frac{g^T B g}{\rho \|g\|^2} + \sqrt{\frac{g^T B g}{\rho \|g\|^2} + \frac{2\|g\|}{\rho}}$
  $\Delta \leftarrow -R_c \frac{g}{\|g\|}$
else
  $\Delta \leftarrow 0, \sigma \leftarrow c' \frac{\sqrt{\epsilon\rho}}{L}, \eta \leftarrow \frac{1}{20L}$
  $\tilde{g} \leftarrow g + \sigma\xi,\ \xi \sim \mathcal{U}\left(\mathbb{S}^{d-1}\right)$
  for $t = 1, \cdots, T(\epsilon)$ **do**
    $\Delta \leftarrow \Delta - \eta(\tilde{g} + B\Delta + \frac{\rho}{2}\|\Delta\|\Delta)$
  end
end

Masiha et al. [2022] have shown that if a weak gradient dominance property holds, SCRN improves the best-known sample complexity of stochastic gradient descent in achieving ϵ-global optimum by a factor of $\mathcal{O}\left(\epsilon^{-1/2}\right)$.

2 Empirical results

For all the RL frames considered in this section, the initial state for the agent is fixed. One could enforce exploration by randomly initialize the agent among all the passable states of the environment: we postpone the discussion about these aspects to future work. To estimate $J(\theta)$ and $\nabla J(\theta)$ given a current parameter value θ, we average over 100 testing policy episodes (with a frequency of 50 episodes during training) under a fixed policy π_θ. The optimum $J(\theta^*)$ is computed assuming a greedy policy for the agent, namely we assume

$$\pi(a|s) = \delta_{a,a^*}, \tag{39}$$

where a^* is the optimal action to be taken in state s. Note that the suboptimality gap

$$J(\theta^*) - J(\theta) \geq 0 \tag{40}$$

will never be zero as a soft-max policy cannot become fully deterministic during training by construction. During all the experiments, a random initialization of the policy is considered. In particular, we always start without any loss of generality from $\theta = 0$, for which the initial policy becomes

$$\pi_\theta(a|s) = \frac{1}{|\mathcal{A}|}, \tag{41}$$

and we compare the performance of SCRN against stochastic policy gradient methods (with and without entropy regularization terms) by looking at how quickly the policy becomes deterministic and how well the algorithms are able to escape saddle points and sub-optimal policies. To study whether (15) holds true with $\epsilon' = 0$ along the trajectories followed by the stochastic methods considered, we estimate the objective function $J(\theta)$ by sampling m trajectories $\{s_t^i, a_t^i\}_{t\geq 0}$, for $1 \leq i \leq m$, followed by the agent under a given fixed policy π_θ. We then define

$$\bar{J}(\theta) := \frac{1}{m} \sum_{j=1}^{m} \sum_{t=0}^{\infty} \gamma^t R(s_t^j, a_t^j) \tag{42}$$

as an unbiased estimate of $J(\theta)$. By adding an entropy bonus to the reward, we define similarly an unbiased estimate of the entropy regularized objective $J^\lambda(\theta)$ defined in (20). In particular, as explained previously, we truncate the infinite sum up to a finite horizon, but this is coherent with the implementation, where we impose a maximum number of steps H per episode. Moreover, the truncation bias can be made arbitrarily small by picking a not too large discount factor γ, which implies that the terms of the form $\gamma^t R(s_t, a_t)$ for $t \geq H+1$ are basically negligible. A similar discussion holds for the estimate of $\nabla J(\theta)$, where we use the same estimate defined at (34) with a large enough m as to guarantee a good approximation of the expected value under the current policy. Upon this definitions, we define the estimated weak gradient dominance constant $\hat{\tau}_J$ as the minimum constant $C_J \in \mathbb{R}$ for which $\bar{J}(\theta) - J(\theta^*) \leq C_J \|\hat{\nabla} J(\theta)\|$ when we consider an estimate of the standard objective (5), and $\bar{J}^\lambda(\theta) - J^\lambda(\theta^*) \leq C_J \|\hat{\nabla} \bar{J}^\lambda(\theta)\|^2$ when we consider the entropy regularized objective (20). This choice is coherent with the discussion we made in the previous sections, as we know that in the deterministic case a weak dominance property holds along the trajectories of policy gradient, while a strong dominance property holds along the trajectories of entropy regularized policy gradient, given an access to the full gradient. Estimating the objective and the gradient by testing the policy π_θ for a sufficient number of episodes comes as a cheaper alternative to the computation of $J(\theta) = V^{\pi_\theta}(s_0)$ by recursively solving the Bellman equation, as doing it during training becomes unnecessarily expensive. Moreover, we also keep track during training of the boundedness away from zero of the action probabilities stated in inequalities (1.2) and (1.3), which we know are nicely bounded away from zero in the deterministic case. To gain further insight on the superior behaviour of SCRN and its ability to escape saddle points, we also look at the eigenvalues of the estimated Hessian.

Hyperparameter setting

For all the environments shown in what follows, the following set of hyperparameters was used for the different algorithms:

- Number of episodes equal to 10000
- Discount factor $\gamma = 0.8$
- For all the algorithms besides the two stages policy gradient, batch sizes for the gradient (and possibly the Hessian) $n_1 = 1$ and $n_2 = 1$
- Learning rate $\alpha = 0.001$ for first order methods, with step decreasing schedule
- Learning rate $\alpha = 1e-4$ for second order methods, with step decreasing schedule
- For entropy regularized objective, the temperature λ was 1
- The estimates (42) and (34) were computed with $m = 100$ and with a test frequency of 50 episodes during training.

Cliff Walking

We start by considering the cliff walking RL task Sutton and Barto [2018]. The agent's aim is to reach a goal state from a start state, avoiding a region of cells called "cliff". The rest of the rectangular state space is free for the agent to move, but to encourage the finding of the shortest path to the goal a negative reward of -0.1 is given in all transitions. The penalty for falling into the cliff is a reward of -100, while reaching the goal on the other side of the cliff gives a reward of +100.

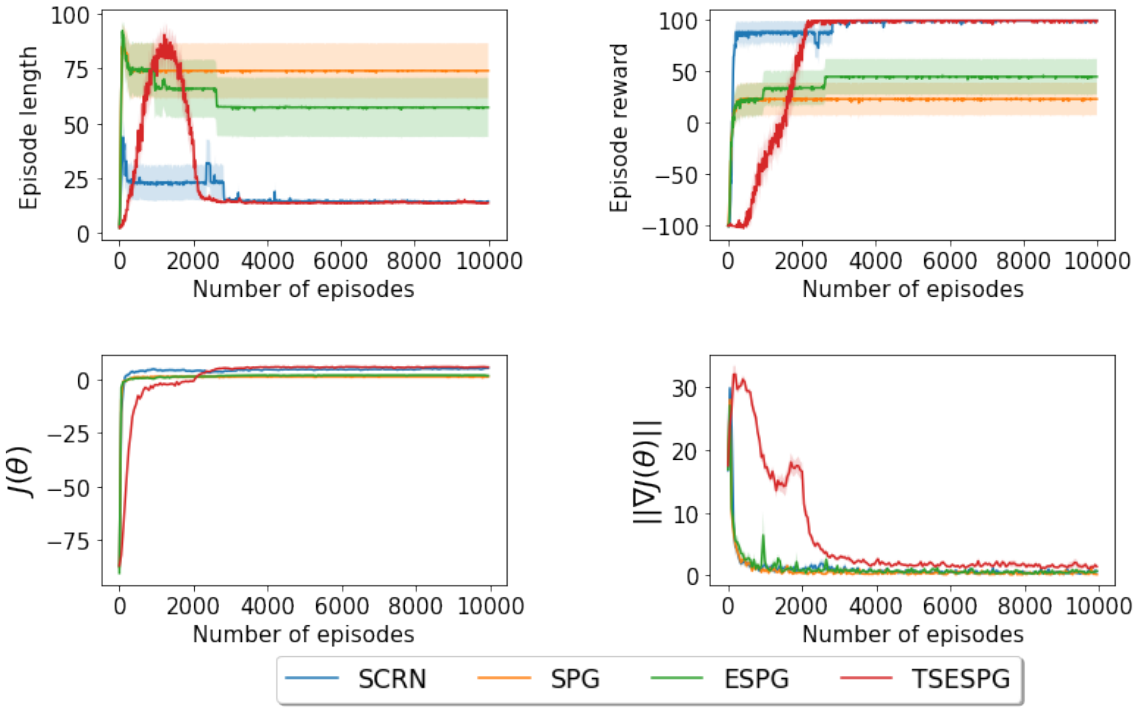

Figure 1: Training results for different RL algorithms in cliff walking optimization. Above: Episode lengths (left) and episode rewards (right) during training for different RL algorithms. Below: objective estimates (left) and gradient estimates (right) during training for different RL algorithms. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

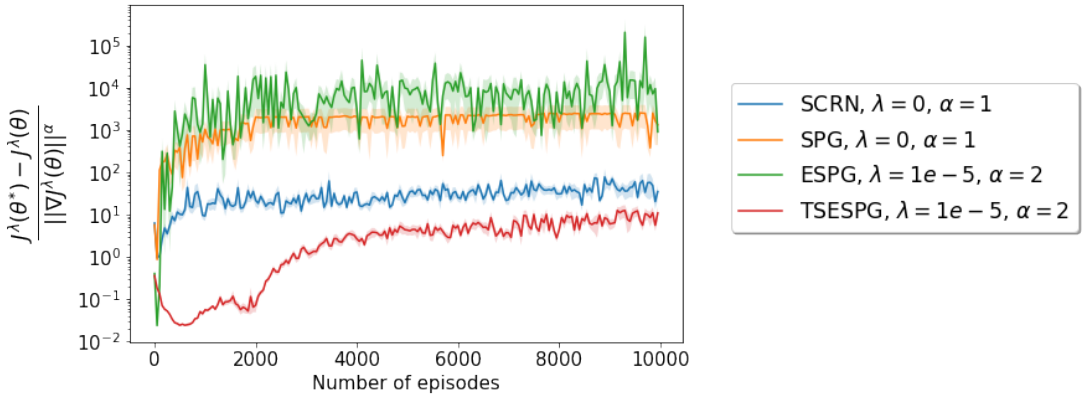

Figure 2: Estimation of τ_J along the trajectories of different RL algorithms for cliff walking optimization. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

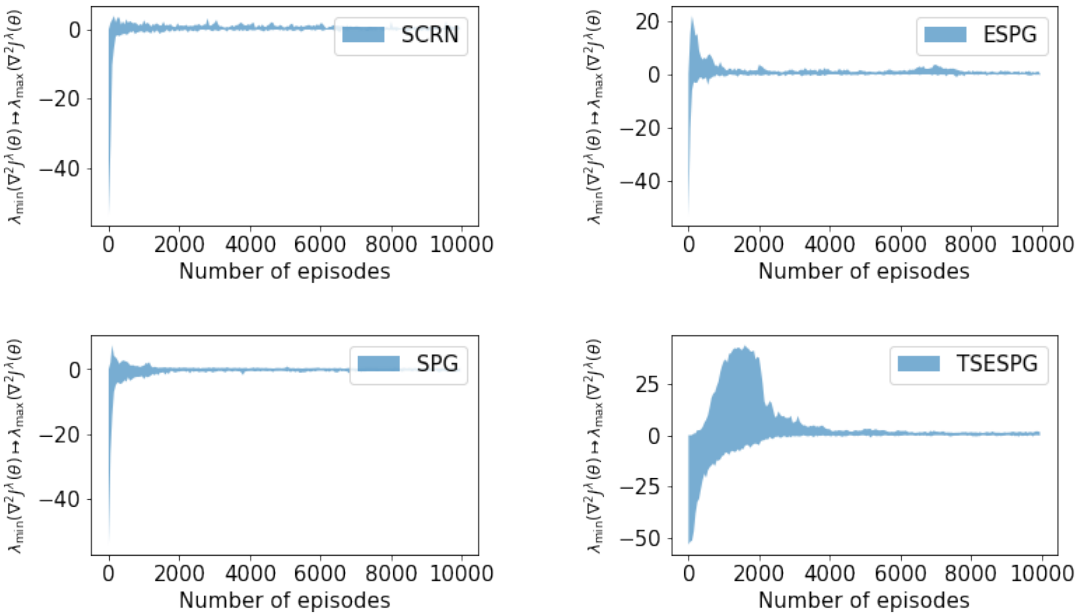

Figure 4: Shaded area between the minimum and the maximum eigenvalue of the Hessian along the trajectories of different RL algorithms. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

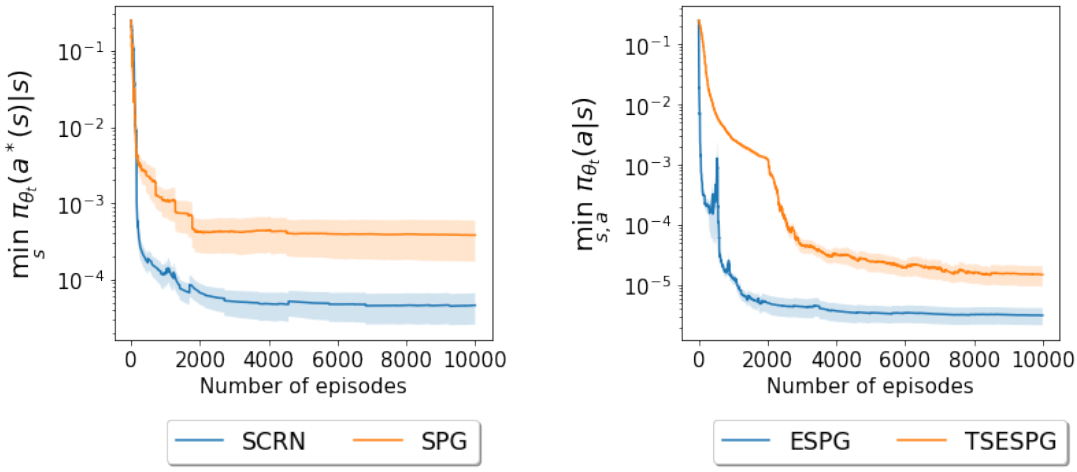

Figure 3: Action probabilities involved in the PL inequalities (19) for the objective (5) and (21) for the objective (20). Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

Figure 1 shows that initially, for all algorithms the episodes have a very short length, because the agent is still unaware of the best action selection and falls into the cliff most of the episodes. The first learning step is actually the avoiding of the cliff, so the agent keeps exploring the state space, now unaware of the highest rewarded state at the end of the cliff region. Then, for SCRN and entropy regularized first order methods, there is a moment of sudden discovery which makes the agent capable of reaching the goal. From this moment, in the remaining episodes the agent mostly tries to shorten the path to the goal by staying as close as possible to the cliff. This happens way sooner during training for SCRN with respect to first order methods. On the other hand, SPG without any entropy bonus fails to become aware of the highest rewarded state and settles for just avoiding the cliff, which yields a suboptimal policy even though the distance from the optimum $J(\theta^*)$ may be contained.

This suggests that while the soft-max policy suffers from the fact that the Fisher information is not strictly positive definite around the optimum $J(\theta^*)$, the preconditioning induced by the full second order information boosts training. Indeed, as pointed out in Masiha et al. [2022], Figure 1 shows that the agent trained with stochastic cubic regularized Newton finds the goal way earlier during the training episodes,

hence with a lower sample complexity. The situation in which the agent just reaches one of the corners and stands there corresponds to a saddle point with a local maximum. However, it is worth noticing that also SCRN is not able to find the optimal path, i.e. to stay always close to the cliff before moving down, but it takes two additional steps because during the first updates it is driven far away from the cliff by having discovered the very negative reward. Hence, this is a relatively easy environment on which we can get a sense of why SCRN works better. To conclude, we make a general remark which holds also for the following environments: under a soft-max policy, computing the Hessian once the gradient is computed does not require much additional computational cost. To see this, recall the estimates for $\nabla J(\theta)$ and $\nabla^2 J(\theta)$ introduced in (8) and (35).

In this environment, Figure 2 shows that the empirical constant τ_J computed along the trajectories followed on average by the algorithms remains nicely bounded from below. Furthermore, it is not diverging when the objective reaches an almost deterministic policy for SCRN and entropy regularized stochastic policy gradient. Instead, it remains more stable and shows less oscillations close to the optimum. We conclude that a weak gradient dominance property holds true, and the constant reaches an asymptotic value as soon as the policy converges, independently of the optimality of the learned policy.

Before moving on to the next environment, we remark that in this case any of the four algorithms found the optimal path. For SCRN, two additional exploration bonuses were attempted, but both have not provided the expected results: re-scaling the soft-max by a temperature parameter β and adding a entropy regularization term similarly to what was done to boost the performance of SPG. The scaling parameter β requires little extra effort in the computation of the gradient and of the Hessian: for a policy

$$\pi_\theta^\beta(a_i|s) = \frac{\beta \exp\{\theta_s^T \mathbb{1}_{a_i}\}}{\sum_j \exp\{\beta \theta_s^T \mathbb{1}_{a_j}\}}, \tag{43}$$

and the corresponding objective $J_\beta(\theta)$, the gradients and the Hessian (and also their unbiased estimates (34) and (36) are just pre-multiplied by β). For β small enough, exploration is vastly encouraged. However, in this particular setting the learned policy was basically independent of β for all optimization methods. A similar discussion holds for entropy regularized SCRN.

Hole discovering

We further considered the hole discovering as a RL task Frémaux et al. [2013]. This is often considered in a bigger class of reinforcement learning tasks as a navigation task, where the goal is hidden behind an obstacle and the trained agent may never become aware of the possibility of getting a very high reward by remaining in the neighbouring of the obstacle. The highest rewarded state (reward +100) is perhaps in the middle of a U-shaped wall. If the agent hits against the wall a reward -100 is given. Again, to encourage a greedy policy towards the goal, for every other action the reward is -0.1.

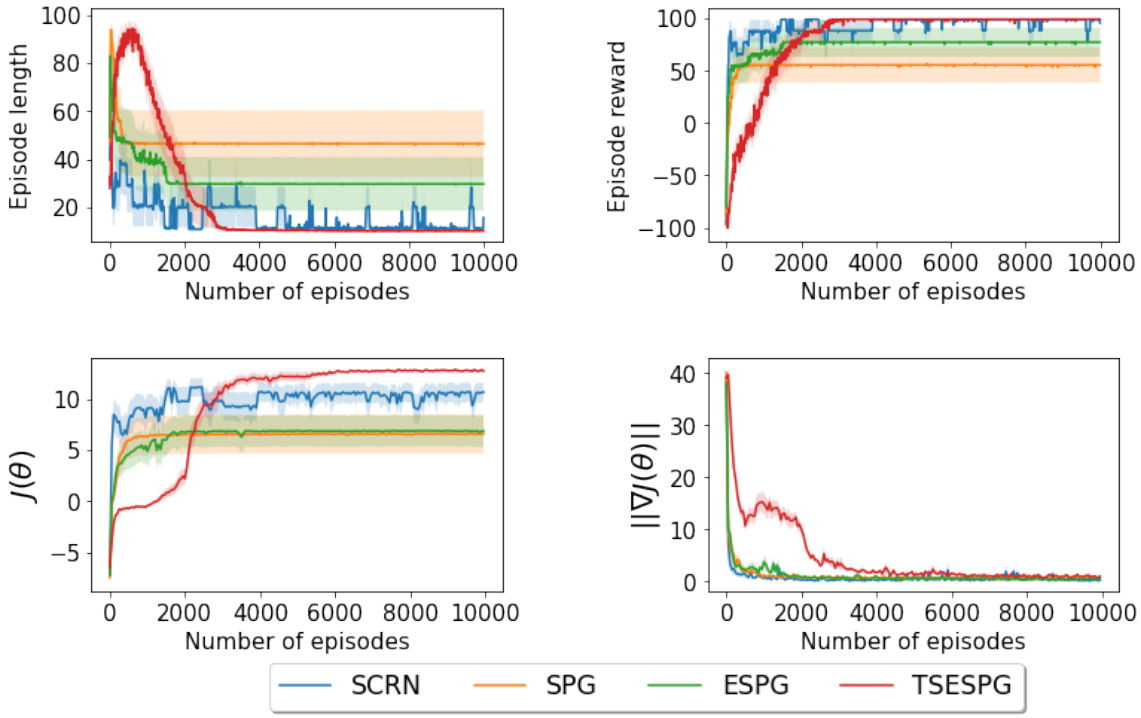

Figure 5: Training results for different RL algorithms in hole discovering optimization. Above: Episode lengths (left) and episode rewards (right) during training for different RL algorithms. Below: objective estimates (left) and gradient estimates (right) during training for different RL algorithms. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

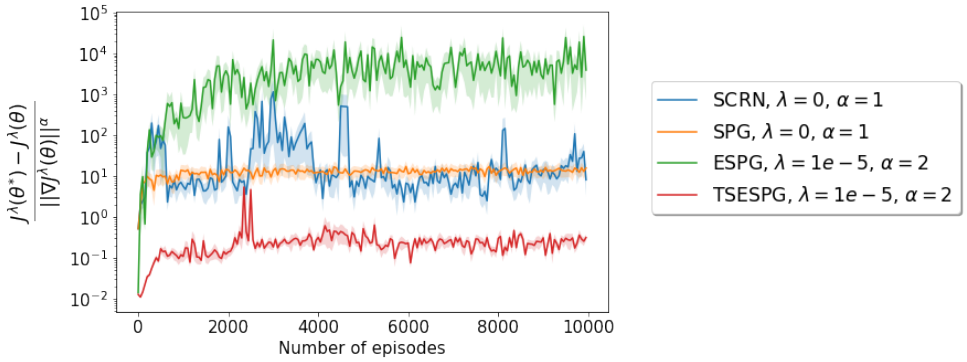

Figure 6: Estimation of τ_J along the trajectories of different RL algorithms for hole discovering optimization. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

Figure 5 shows that, contrarily to the cliff walking optimization shown in Figure 1, all algorithms are finding a way to reach the goal state. However, once again, SCRN finds the shortest path in the smaller number of episodes, while for SPG and entropy regularized SPG the agent cannot find a short path to the goal on average. As a further difference with the above, for all the algorithms the agent is able to capture immediately the presence of a wall in the middle of the state space. It then takes a little while to realize that in the middle it can find also the highest rewarded state. Figure 5 shows that, while SCRN is still outperforming the other algorithms in terms of quickness in finding the shortest path to the goal, close to the optimum it exhibits some oscillatory and unstable behaviour. On the other hand, the two stages entropy regularized SPG algorithm shows more stability after reaching a greedy policy which leads every time to the goal state. Still, Figure 6 shows that the constant τ_J remains bounded from above and reaches an asymptotic value, without exploding, when the algorithms reach a greedy policy (optimal or not).

Random shape maze

We further considered the random mazes as a RL task Zuo [2018]. The goal in this case is to find the goal (reward +100) at the end of an obstacle sequence which is random at every training instance. If the agent hits the obstacles a reward −100 is given, for every other action the reward is −0.1. Figure 8 shows that also in this case there seems to be a correlation between maintaining a low τ_J constant and policy convergence: stochastic policy gradient again fails to find the optimal path while SCRN finds it and finds the optimal path (i.e. it reaches the goal state in the minimum number of steps possible). In this environment, it is the only algorithm consistently capable of finding the goal, as also the two stages regularized SPG fails in this case and just escapes the obstacles on the state space rather than finding the highest rewarded state. In particular, while we know that Fisher non-degeneracy is violated as the softmax policy converges to a deterministic one, a weak gradient dominance property seems to hold without any diverging behaviour. Hence, empirically (15) holds even if we get arbitrarily close to the maximum of $J(\theta)$. Moreover, while we observe once more the superior behaviour of SCRN induced by the second order information, we recognize again an unstable behaviour close to the optimum.

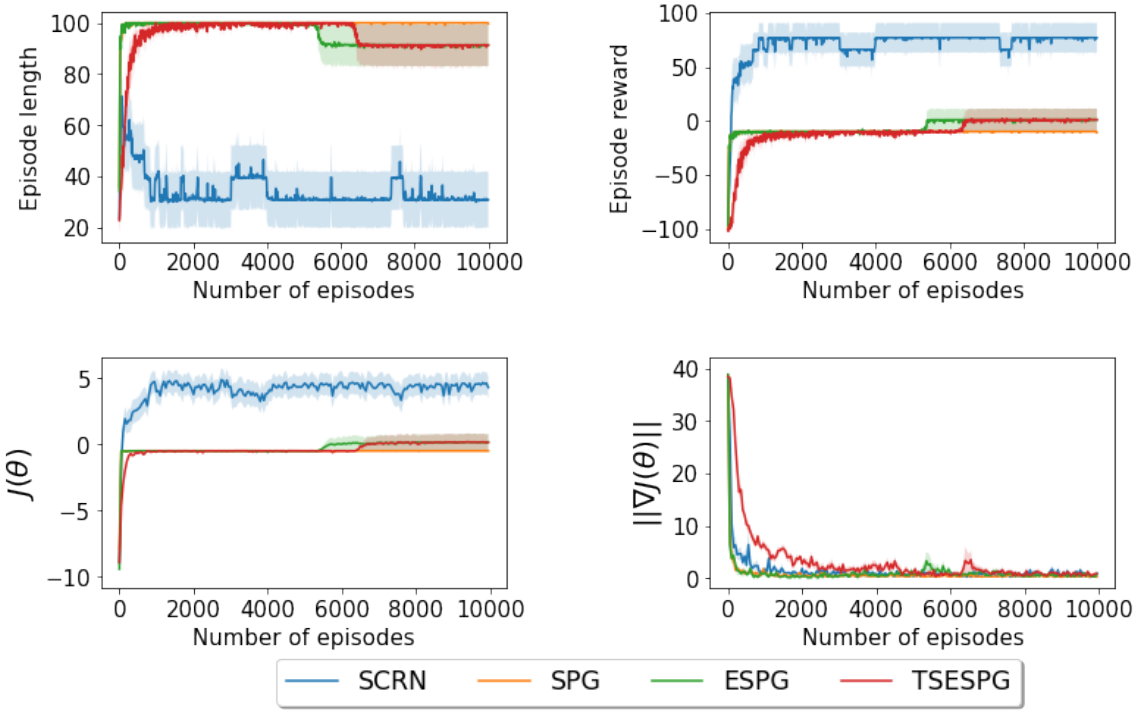

Figure 7: Training results for different RL algorithms in random shape maze optimization. Above: Episode lengths (left) and episode rewards (right) during training for different RL algorithms in random maze optimization. Below: objective estimates (left) and gradient estimates (right) during training. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

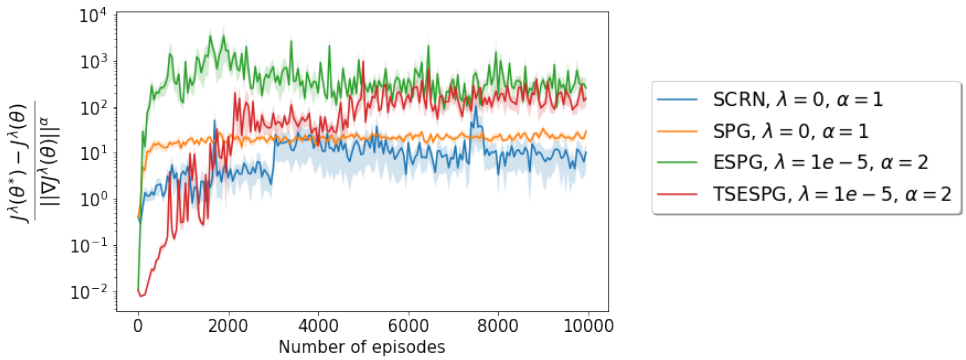

Figure 8: Estimation of τ_J along the trajectories of different RL algorithms for random mazes optimization. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

U Maze

We further considered the U shaped mazes as a RL task Zuo [2018]. The goal in this case is to find the goal (reward +100) on the other side of a U-shaped maze, where the obstacle is represented by the area which is not touched by the U. If the agent hits such a obstacle a reward −100 is given, for every other action the reward is −0.1 to encourage the agent to find the optimal path as well. Figure 10 shows that stochastic policy gradient with and without entropy regularization are both failing to find the highest rewarded state while SCRN finds a good path (but not optimal) to the goal, and a similar discussion holds true for the two stages entropy regularized SPG. In particular, while we know that Fisher non-degeneracy is violated as the softmax policy converges to a deterministic one, a weak gradient dominance property seems to hold without any diverging behaviour. Hence, we observe once again that empirically (15) holds even if we get arbitrarily close to the maximum of $J(\theta)$.

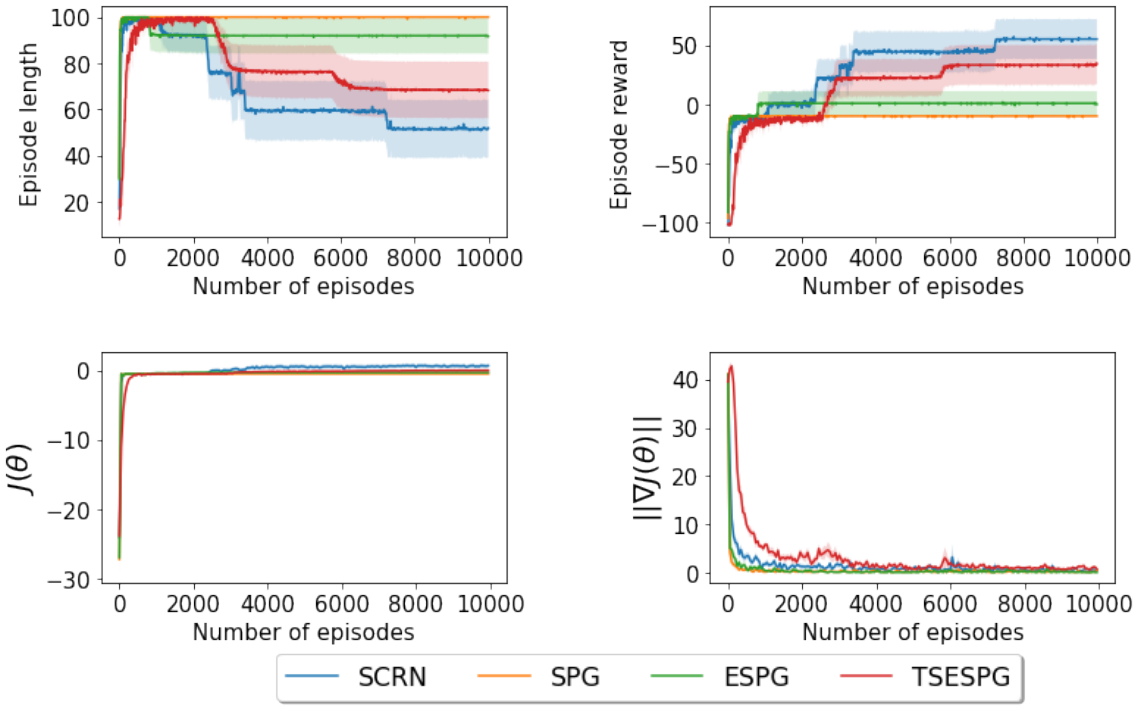

Figure 9: Training results for different RL algorithms in U-shaped maze optimization. Above: Episode lengths (left) and episode rewards (right) during training for different RL algorithms in random maze optimization. Below: objective estimates (left) and gradient estimates (right) during training. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

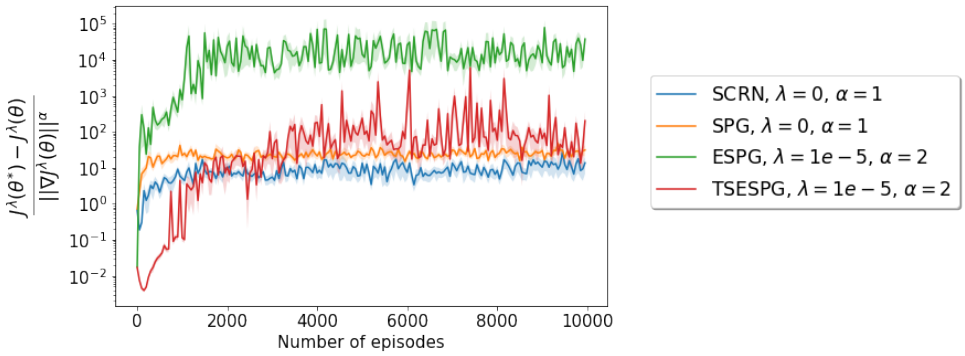

Figure 10: Estimation of τ_J along the trajectories of different RL algorithms for U-shaped maze optimization. Results are averaged over 10 full training runs. Shaded regions show the standard deviation.

3 A comparison between CRN and NPG

Natural policy gradient has been the starting point for the theoretical development of many successful algorithms in policy gradient's based reinforcement learning such as TRPO (Trust Region Policy Optimization) Schulman et al. [2015] and PPO (Proximal Policy Optimization) Schulman et al. [2017]. Natural policy gradient belongs to a general class of update rule of the form

$$\theta_t \leftarrow \theta_t + \alpha_t (M(\theta_t))^{-1} \nabla J(\theta_t), \tag{44}$$

where $M(\theta)$ is a preconditioner for the gradient which needs to be chosen in a trade-off fashion between computational cost and performance in revealing information about the landscape of the objective function, which is for us the usual expected return $J(\theta)$ defined in (5). The standard policy gradient rule takes naively $M(\theta_t) = I$ as the identity matrix for all $t \geq 0$, and hence no preconditioning is applied. Considering the second order information given by $\nabla^2 J(\theta)$, one recovers the Newton's Method which we briefly introduced above. Natural Policy Gradient chooses the Fisher information matrix defined in (13) as a preconditioner for the update. It is clear from the form of (44) that for such an algorithm to work well, Fisher non-degeneracy must hold true such that the inverse of $M(\theta_t)$ is always well defined. However, as we extensively clarified throughout the elaborate, for a soft-max policy, the Fisher information matrix becomes degenerate when the algorithm gets arbitrarily close to a greedy policy, whether the latter is optimal or not. This makes it difficult to study the performance of such an algorithm under a soft-max policy. In this section, we aim at a first comparison between the second order information for preconditioning the gradient update when using the whole Hessian matrix and the Fisher information (13), or estimates of the latter quantities. We then reproduce the results provided in the pioneering work about NPG methods Kakade [2001] and we discuss further improvements of the current analysis. Let us start by doing a brief recap on the motivation of natural policy gradients.

Natural Policy Gradient

The update (9) does not take into account the possibility of overshooting, and for an RL setting this may result in a poorly updated policy from which the agent is not able to recover, because future sample batches obtained by interacting with the environment may not provide much meaningful information. A naive solution to prevent overshooting is to pick a small enough learning rate α, but this slows down convergence. As an alternative approach, **natural policy gradient** methods clip the change in the policy instead of the change in parameters. Traditional policy gradient methods are fundamentally flawed. Natural gradients converge quicker and better, forming the foundation of contemporary Reinforcement Learning such as Trust Region Policy Optimization (TRPO) and Proximal Policy Optimization (PPO). In traditional policy gradient methods, the gradient $\nabla J(\theta)$ only gives the direction of the weight update. It does not tell how far to step into that direction. Derivatives are defined on an infinitesimal interval, meaning the gradient is valid only locally, and may be completely different at another part of the function (5). Because of this, we iterate between sampling (with the current policy) and updating (based on sampled data). Each new policy rollout allows recomputing the gradient and update the policy weights θ. Behavior is controlled with the step size α. as we did in (34). The update (9) does not take into account the possibility of overshooting, and for an RL setting this may result in a poorly updated policy from which the agent is not able to recover, because future sample batches obtained by interacting with the environment may not provide much meaningful information. A very naive solution may be to pick a small learning rate α, but this often results in slow convergence. Another possibility would be to cap the update changes to some ϵ tolerance, namely to define

$$\Delta\theta = \text{argmax}_{\|\Delta\theta\| \leq \epsilon} J(\theta + \Delta\theta), \tag{45}$$

but capping the parameter space does not effectively cap the statistical (and nonlinear) manifold of random policies on which the algorithm is operating. To prevent overshooting during an update, one should impose a constraint on the policy change instead of the parameters. Capping the difference between the old policy and the updated one is a much more powerful way to control the optimization problem on the statistical manifold of policies. This is precisely the main advantage of second order information: by preconditioning the gradient with curvature information capturing the sensitivity of the manifold to marginal parameter changes, we ensure that the old and the new policy are not too far from each other, and this on one hand prevents overshooting and on the other hand makes learning faster in flat, suboptimal plateaus where first order methods as policy gradients may remain stuck.

To define the distance between two policies (i.e. probability distributions defined on the state-action space of a RL environment), the **Kullback-Leibner divergence** is often considered. The latter is

defined as

$$\mathcal{D}_{KL}(\pi_\theta \| \pi_{\theta+\Delta\theta}) = \mathbb{E}_{\pi_\theta}\left[\log\left(\frac{\pi_\theta}{\pi_{\theta+\Delta\theta}}\right)\right]. \tag{46}$$

The locally defined KL divergence is actually equivalent with the Fisher information matrix (13). The Fisher information matrix is often defined as the Riemannian metric describing the curvature of a statistical manifold, and it may be viewed as a correction to the distance that accounts for the curvature. If the Fisher matrix is an identity matrix, the distance over the manifold is simply the Euclidean distance. In that case, traditional and natural policy gradients are equivalent.

If we now modify (45) with a constraint on the policy change rather than the parameter change, we may define the local optimization problem as

$$\Delta\theta = \text{argmax}_{\mathcal{D}_{KL}(\pi_\theta \| \pi_{\theta+\Delta\theta}) \leq \epsilon} J(\theta + \Delta\theta). \tag{47}$$

However, computing the KL divergence for large RL environments is often not feasible. This is where the Fisher information matrix becomes a more appealing solution. We start by solving the unconstrained optimization problem

$$\Delta\theta^* = \text{argmax}_{\Delta\theta} J(\theta + \Delta\theta) - \lambda(\mathcal{D}_{KL}(\pi_\theta \| \pi_{\theta+\Delta\theta}) - \epsilon), \tag{48}$$

as a surrogate (and easier to handle) objective with respect to (47). By Taylor expansion in a neighbourhood of the current policy θ we get

$$\Delta\theta^* \sim \text{argmax}_{\Delta\theta} J(\theta) + \nabla J(\theta)^T \Delta\theta - \frac{1}{2}\lambda(\Delta\theta^T \nabla^2 \mathcal{D}_{KL}(\pi_\theta \| \pi_{\theta+\Delta\theta})\Delta\theta) + \lambda\epsilon.$$

Because we are considering the local KL divergence, we may substitute the Hessian of the KL divergence by the Fisher information matrix. The Fisher information matrix can be represented as the outer product of policy gradients. The expression is locally equivalent to the Hessian matrix of the KL divergence, but computationally more efficient to generate. Moreover, let us drop all the terms which are not depending on $\Delta\theta$ explicitly. This gives

$$\Delta\theta^* \sim \nabla J(\theta)^T \Delta\theta - \frac{1}{2}\lambda \Delta\theta^T F_\rho(\theta)\Delta\theta.$$

Now, we aim at maximizing the parameter change while we can use the Lagrangian multiplier λ to impose a small change in the policy. If Fisher non-degeneracy holds, it is straightforward to verify that the objective $\nabla J(\theta)^T \Delta\theta - \frac{1}{2}\lambda\Delta\theta^T F_\rho(\theta)\Delta\theta$ is strongly concave with a unique maximizer given by $\Delta\theta^* = -\frac{1}{\lambda} F_\rho(\theta)^{-1} \nabla J(\theta)$. λ is then used to impose a dynamical learning rate satisfying the KL constraint. Finally, this gives the update

$$\theta \leftarrow \theta + \alpha F_\rho(\theta)^{-1} \nabla J(\theta), \tag{49}$$

which is the standard gradient preconditioned with the inverse Fisher matrix, accounting for the curvature of the Riemannian space.

Fisher information matrix and Hessian matrix

Let us begin this comparison by recalling the expression for the Fisher information matrix

$$F_\rho(\theta) = \mathbb{E}_{(s,a)\sim\nu_\rho^{\pi_\theta}(s,a)}\left[\nabla \log \pi_\theta(a|s) \nabla \log \pi_\theta(a|s)^T\right], \tag{50}$$

for the quantities defined previously. From Theorem 3 in Furmston et al. [2016], we know that we can write the Hessian of the objective $J(\theta)$ as

$$\begin{aligned}\nabla^2 J(\theta) = \mathbb{E}_{\tau\sim p(\cdot|\pi_\theta)}&\left[\sum_{h=0}^{H-1}\left(\sum_{t=h}^{H-1}\gamma^t R(s_t,a_t)\right)\nabla\log\pi_\theta(a_h|s_h)\nabla\log\pi_\theta(a_h|s_h)^T\right] \\ &+\left[\sum_{h=0}^{H-1}\left(\sum_{t=h}^{H-1}\gamma^t R(s_t,a_t)\right)\nabla^2\log\pi_\theta(a_h|s_h)\right].\end{aligned} \tag{51}$$

In the first term of (51), we recognize an expression similar to that of the Fisher information matrix, but weighted with the state-action value function $\sum_{t=h}^{H-1}\gamma^t R(s_t,a_t)$. Indeed, while in the Fisher information matrix the expectation is taken with respect to the geometrically weighted summation of state-action occupancy marginals of the trajectory distribution, in the first term of the Hessian there is an additional

weighting from the state-action value function. Hence, the latter expression incorporates information about the reward structure of the objective function, and hence about the curvature of the objective function. The second term in (51) is the Hessian of the log-likelihood of the policy, i.e. the gradient of the score function scaled by its value in expectation with respect to the discounted state-occupancy measure over the state-space $\mathcal{S}$. Interestingly, it has been shown empirically in the hallway problem in Furmston et al. [2016] that when getting closer to the optimal policy, the first part of the Hessian in (51) tends to dominate the second term in spectral norm.

The case of a simple MDP

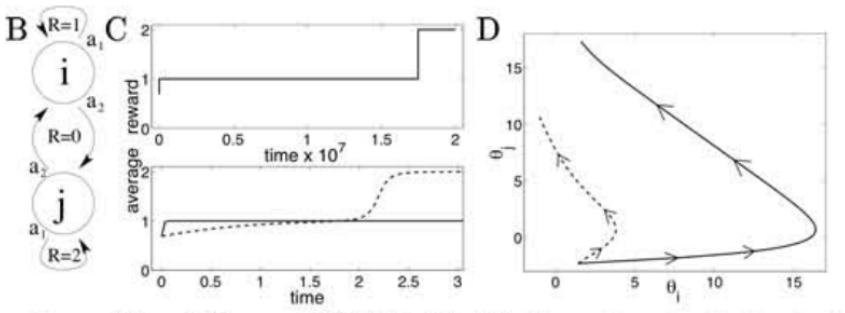

Figure 11: Simple MDP which gives insight on the superior behaviour of NPG over SPG. The agent always starts in state i and its goal is to reach state j. However, a self-loop from state i gives a reward of 1, which is a suboptimal trap in which the agent must not fall. Indeed, a self-loop from state j gives a higher reward equal to 2.

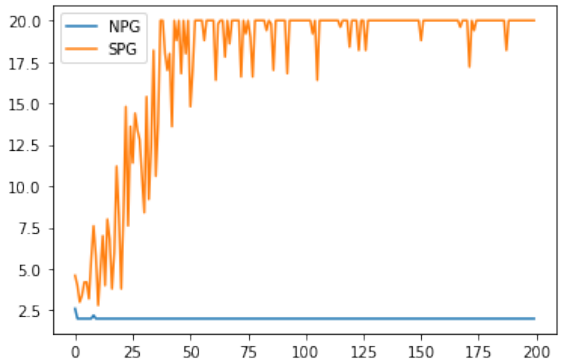

Figure 12: Results for NPG and SPG on the simple MDP shown in Figure 11. We see that SPG is stuck in state i throughout all the episodes, while NPG takes only 3 episodes to go straight to the highest rewarded state.

4 Conclusions

References

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. Optimality and approximation with policy gradient methods in markov decision processes. In *Conference on Learning Theory*, pages 64–66. PMLR, 2020.

A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *J. Mach. Learn. Res.*, 22(98):1–76, 2021.

R. Battiti. First-and second-order methods for learning: between steepest descent and newton's method. *Neural computation*, 4(2):141–166, 1992.

Y. Ding, J. Zhang, and J. Lavaei. Beyond exact gradients: Convergence of stochastic soft-max policy gradient methods with entropy regularization. *arXiv preprint arXiv:2110.10117*, 2021.

N. Frémaux, H. Sprekeler, and W. Gerstner. Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS computational biology*, 9(4):e1003024, 2013.

T. Furmston, G. Lever, and D. Barber. Approximate newton methods for policy search in markov decision processes. *Journal of Machine Learning Research*, 17, 2016.

Y. Grandvalet and Y. Bengio. Entropy regularization., 2006.

S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer, 2002.

S. M. Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Y. Liu, K. Zhang, T. Basar, and W. Yin. An improved analysis of (variance-reduced) policy gradient and natural policy gradient methods. *Advances in Neural Information Processing Systems*, 33:7624–7636, 2020.

S. Masiha, S. Salehkaleybar, N. He, N. Kiyavash, and P. Thiran. Stochastic second-order methods provably beat sgd for gradient-dominated functions. *arXiv preprint arXiv:2205.12856*, 2022.

J. Mei, C. Xiao, C. Szepesvari, and D. Schuurmans. On the global convergence rates of softmax policy gradient methods. In *International Conference on Machine Learning*, pages 6820–6829. PMLR, 2020.

Y. Nesterov and B. T. Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Z. Shen, A. Ribeiro, H. Hassani, H. Qian, and C. Mi. Hessian aided policy gradient. In *International conference on machine learning*, pages 5729–5738. PMLR, 2019.

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

N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. I. Jordan. Stochastic cubic regularization for fast nonconvex optimization. *Advances in neural information processing systems*, 31, 2018.

R. Yuan, R. M. Gower, and A. Lazaric. A general sample complexity analysis of vanilla policy gradient. In *International Conference on Artificial Intelligence and Statistics*, pages 3332–3380. PMLR, 2022.

X. Zuo. mazelab: A customizable framework to create maze and gridworld environments, 2018.