Abstract

In this paper, we propose a novel reinforcement-learning algorithm consisting in a stochastic variance-reduced version of policy gradient for solving Markov Decision Processes (MDPs). Stochastic variance-reduced gradient (SVRG) methods have proven to be very successful in supervised learning. However, their adaptation to policy gradient is not straightforward and needs to account for I) a non-concave objective function; II) approximations in the full gradient computation; and III) a non-stationary sampling process. The result is SVRPG, a stochastic variance-reduced policy gradient algorithm that leverages on importance weights to preserve the unbiasedness of the gradient estimate. Under standard assumptions on the MDP, we provide convergence guarantees for SVRPG with a convergence rate that is linear under increasing batch sizes. Finally, we suggest practical variants of SVRPG, and we empirically evaluate them on continuous MDPs.

1. Introduction

On a very general level, artificial intelligence addresses the problem of an agent that must select the right actions to solve a task. The approach of Reinforcement Learning (RL) (Sutton & Barto, 1998) is to learn the best actions by direct interaction with the environment and evaluation of the performance in the form of a reward signal. This makes RL fundamentally different from Supervised Learning (SL), where correct actions are explicitly prescribed by a human teacher (e.g., for classification, in the form of class labels). However, the two approaches share many challenges and tools. The problem of estimating a model from samples, which is at the core of SL, is equally fundamental in RL, whether we choose to model the environment, a value function, or directly a policy defining the agent’s behaviour. Furthermore, when the tasks are characterized by large or continuous state-action spaces, RL needs the powerful function approximators (e.g., neural networks) that are the main subject of study of SL. In a typical SL setting, a performance function $J(\theta)$ has to be optimized w.r.t. to model parameters $\theta$. The set of data that are available for training is often a subset of all the cases of interest, which may even be infinite, leading to optimization of finite sums that approximate the expected performance over an unknown data distribution. When generalization to the complete dataset is not taken into consideration, we talk about Empirical Risk Minimization (ERM). Even in this case, stochastic optimization is often used for reasons of efficiency. The idea of stochastic gradient (SG) ascent (Nesterov, 2013) is to iteratively focus on a random subset of the available data to obtain an approximate improvement direction. At the level of the single iteration, this can be much less expensive than taking into account all the data. However, the sub-sampling of data is a source of variance that can potentially compromise convergence, so that per-iteration efficiency and convergence rate must be traded off with proper handling of meta-parameters. Variance-reduced gradient algorithms such as SAG (Roux et al., 2012), SVRG (Johnson & Zhang, 2013) and SAGA (Defazio et al., 2014a) offer better ways of solving this trade-off, with significant results both in theory and practice. Although designed explicitly for ERM, these algorithms address a problem that affects more general machine learning problems.

In RL, stochastic optimization is rarely a matter of choice, since data must be actively sampled by interacting with an initially unknown environment. In this scenario, limiting the variance of the estimates is a necessity that cannot be avoided, which makes variance-reduced algorithms very interesting. Among RL approaches, policy gradient (Sutton et al., 2000) is the one that bears the closest similarity to SL solutions. The fundamental principle of these methods is to optimize a parametric policy through stochastic gradient ascent. Compared to other applications of SG, the cost of collecting samples can be very high since it requires to interact with the environment. This makes SVRG-like methods potentially much more efficient than, e.g., batch learning. Unfortunately, RL has a series of difficulties that are not present in ERM. First, in SL the objective can often be de-
A Reinforcement Learning task (Sutton & Barto, 1998) can be modelled with a discrete-time continuous Markov Decision Process (MDP) \( M =\{S,A,P,R,\gamma,\rho\} \), where \( S \) is a continuous state space; \( A \) is a continuous action space; \( P \) is a Markovian transition model, where \( P(s'|s,a) \) defines the transition density from state \( s \) to \( s' \) under action \( a \); \( R \) is the reward function, where \( R(s,a) \in [-R, R] \) is the expected reward for state-action pair \( (s,a) \); \( \gamma \in [0,1) \) is the discount factor; and \( \rho \) is the initial state distribution. The agent’s behaviour is modelled as a policy \( \pi \), where \( \pi(\cdot|s) \) is the density distribution induced by policy \( \pi \) on the set \( T \) of all possible trajectories (see Appendix A for the definition), and with \( R(\tau) \) the total discounted reward provided by trajectory \( \tau: R(\tau) = \sum_{t=0}^{H-1} \gamma^t R(s_t,a_t) \). Policies can be ranked based on their expected total reward: \( J(\pi) = \mathbb{E}_{\tau \sim \pi(\cdot|\pi)}[R(\tau)|M] \). Solving an MDP \( M \) means finding \( \pi^* \in \arg \max_{\pi} J(\pi) \).

Policy gradient methods restrict the search for the best performing policy over a class of parametrized policies \( \Pi_\theta = \{\pi_\theta : \theta \in \mathbb{R}^d\} \), with the only constraint that \( \pi_\theta \) is differentiable w.r.t. \( \theta \). For sake of brevity, we will denote the performance of a parametric policy with \( J(\theta) \) and the probability of a trajectory \( \tau: p(\tau|\theta) \) (in some occasions, \( p(\tau|\theta) \) will be replaced by \( p_\theta(\tau) \) for the sake of readability). The search for a locally optimal policy is performed through gradient ascent, where the policy gradient is (Sutton et al., 2000; Peters & Schaal, 2008a):

\[
\nabla J(\theta) = \mathbb{E}_{\tau \sim p(\cdot|\theta)} \left[ \nabla \log p_\theta(\tau) R(\tau) \right].
\]

Notice that the distribution defining the gradient is induced by the current policy. This aspect introduces a non-stationarity in the sampling process. Since the underlying distribution changes over time, it is necessary to resample at each update or use weighting techniques such as importance sampling. Here, we consider the online learning scenario, where trajectories are sampled by interacting with the environment at each policy change. In this setting, stochastic gradient ascent is typically employed. At each iteration \( k \geq 0 \), a batch \( D_N^k = \{\tau_i\}_{i=0}^N \) of \( N > 0 \) trajectories is collected using policy \( \pi_{\theta_k} \). The policy is then updated as \( \theta_{k+1} = \theta_k + \alpha \nabla_{\theta_k} J(\theta_k) \), where \( \alpha \) is a step size and \( \nabla_{\theta_k} J(\theta_k) \) is an estimate of Eq. (1) using \( D_N^k \). The most common policy gradient estimators (e.g., REINFORCE (Williams, 1992) and G(PO)MDP (Baxter & Bartlett, 2001)) can be expressed as follows

\[
\nabla_{\theta_k} J(\theta_k) = \frac{1}{N} \sum_{n=1}^N g(\tau_i|\theta_k), \quad \tau_i \in D_N^k,
\]

where \( g(\tau_i|\theta_k) \) is an estimate of \( \nabla \log p_\theta(\tau_i) R(\tau_i) \). Although the REINFORCE definition is simpler than the G(PO)MDP one, the latter is usually preferred due to its lower variance. We refer the reader to Appendix A for details and a formal definition of \( g \).

The main limitation of plain policy gradient is the high variance of these estimators. The naive approach of increasing the batch size is not an option in RL due to the high cost of collecting samples, i.e., by interacting with the environment. For this reason, literature has focused on the introduction of baselines (i.e., functions \( b: S \times A \rightarrow \mathbb{R} \)) aiming to reduce the variance (e.g., Williams, 1992; Peters & Schaal, 2008a).
2.2. Stochastic Variance-Reduced Gradient

Finite-sum optimization is the problem of maximizing an objective function \( f(\theta) \) which can be decomposed into the sum or average of a finite number of functions \( z_i(\theta) \):

\[
\max_{\theta} \left\{ f(\theta) = \frac{1}{N} \sum_{i=1}^{N} z_i(\theta) \right\}.
\]

This kind of optimization is very common in machine learning, where each \( z_i \) may correspond to a data sample \( x_i \) from a dataset \( D_N \) of size \( N \) (i.e., \( z_i(\theta) = z(x_i|\theta) \)). A common requirement is that \( z \) must be smooth and concave in \( \theta \).

Under this hypothesis, full gradient (FG) ascent (Cauchy, 1847) with a constant step size achieves a linear convergence rate in the number \( T \) of iterations (i.e., parameter updates) (Nesterov, 2013). However, each iteration requires \( N \) gradient computations, which can be too expensive for large values of \( N \). Stochastic Gradient (SG) ascent (e.g., Robbins & Monro, 1951; Bottou & LeCun, 2004) overcomes this problem by sampling a single sample \( x_i \) per iteration, but a vanishing step size is required to control the variance introduced by sampling. As a consequence, the lower per-iteration cost is paid with a worse, sub-linear convergence rate (Nemirovskii et al., 1983). Starting from SAG, a series of variations to SG have been proposed to achieve a better trade-off between convergence speed and cost per iteration: e.g., SAG (Roux et al., 2012), SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014a), Finito (Defazio et al., 2014b), and MISO (Mairal, 2015). The common idea is to reuse past gradient computations to reduce the variance of the current estimate. In particular, Stochastic Variance-Reduced Gradient (SVRG) is often preferred to other similar methods for its limited storage requirements, which is a significant advantage when deep and/or wide neural networks are employed.

Algorithm 1 SVRG

**Input:** a dataset \( D_N \), number of epochs \( S \), epoch size \( m \), step size \( \alpha \), initial parameter \( \theta^0 \)

**for** \( s = 0 \) to \( S - 1 \) **do**

\[
\theta^s_m := \theta^s
\]

\[
\bar{\mu} = \nabla f(\theta^s)
\]

**for** \( t = 0 \) to \( m - 1 \) **do**

\[
x \sim \mathcal{U}(D_N)
\]

\[
v_{t+1} = \bar{\mu} + \nabla z(x|\theta^s) - \nabla z(x|\theta^s)
\]

\[
\theta^s_{t+1} = \theta^s_{t+1} + \alpha v_{t+1}
\]

**end for**

**end for**

**Concave case:** return \( \theta^s_m \)

**Non-Concave case:** return \( \theta^s_{t+1} \) with \((s,t)\) picked uniformly at random from \([0, S - 1] \times [0, m - 1]\)

The idea of SVRG (Algorithm 1) is to alternate full and stochastic gradient updates. Each \( m = O(N) \) iterations, a snapshot \( \theta^s \) of the current parameter is saved together with its full gradient \( \nabla f(\theta^s) = \frac{1}{N} \sum z(\theta^s) \). Between snapshots, the parameter is updated with \( \nabla f(\theta) \), a gradient estimate corrected using stochastic gradient. For any \( t \in \{0, \ldots, m - 1\} \):

\[
\nabla f(\theta_t) := v_t = \nabla f(\bar{\theta}) + \nabla z(x|\theta_t) - \nabla z(x|\bar{\theta}),
\]

where \( x \) is sampled uniformly at random from \( D_N \) (i.e., \( x \sim \mathcal{U}(D_N) \)). Note that \( t = 0 \) corresponds to a FG step (i.e., \( \nabla f(\theta^0) = \nabla f(\bar{\theta}) \)) since \( \theta^0 := \bar{\theta} \). The corrected gradient \( \nabla f(\theta) \) is an unbiased estimate of \( \nabla f(\theta) \), and it is able to control the variance introduced by sampling even with a fixed step size, achieving a linear convergence rate without resorting to a plain full gradient.

More recently, some extensions of variance reduction algorithms to the non-concave objectives have been proposed (e.g., Allen-Zhu & Hazan, 2016; Reddi et al., 2016a,b). In this scenario, \( f \) is typically required to be \( L \)-smooth, i.e., \( \|
abla f(\theta') - \nabla f(\theta)\|_2 \leq L \|	heta' - \theta\|_2 \) for each \( \theta, \theta' \in \mathbb{R}^n \) and for some Lipschitz constant \( L \). Under this hypothesis, the convergence rate of SG is \( O(1/\sqrt{T}) \) (Ghadimi & Lan, 2013), i.e., \( T = O(1/\epsilon^2) \) iterations are required to get \( \|
abla f(\theta)\|_2 \leq \epsilon \). Again, SVRG achieves the same rate as FG (Reddi et al., 2016a), which is \( O(1/k) \) in this case (Nesterov, 2013). The only additional requirement is to select \( \theta^* \) uniformly at random among all the \( \theta_k \) instead of simply setting it to the final value \( k \) being the iterations.

3. SVRG in Reinforcement Learning

In online RL problems, the usual approach is to tune the batch size of SG to find the optimal trade-off between variance and speed. Recall that, compared to SL, the samples...
We can now introduce Stochastic Variance-Reduced Policy Gradient (SVRG), which is a reasonable assumption for common policy classes. (2017) performed a similar analysis on MISO. In Section 4 Appendix A).

Non-concavity: the objective function $J(\theta)$ is typically non-concave.

Infinite dataset: the RL optimization cannot be expressed as a finite-sum problem. The objective function is an expected value over the trajectory density $p_{\theta}(\tau)$ of the total discounted reward, for which we would need an infinite dataset.

Non-stationarity: the distribution of the samples changes over time. In particular, the value of the policy parameter $\theta$ influences the sampling process.

To deal with non-concavity, we require $J(\theta)$ to be $L$-smooth, which is a reasonable assumption for common policy classes such as Gaussian\(^3\) and softmax (e.g., Furmston & Barber, 2012; Pirotta et al., 2015). Because of the infinite dataset, we can only rely on an estimate of the full gradient. Harikandeh et al. (2015) analysed this scenario under the assumptions of $\tau$ being concave, showing that SVRG is robust to an inexact computation of the full gradient. In particular, it is still possible to recover the original convergence rate if the error decreases at an appropriate rate. Bietti & Mairal (2017) performed a similar analysis on MISO. In Section 4 we will show how the estimation accuracy impacts on the convergence results with a non-concave objective. Finally, the non-stationarity of the optimization problem introduces a bias into the SVRG estimator in Eq. (3). To overcome this limitation we employ importance weighting (e.g., Rubin, 1981; Precup, 2000) to correct the distribution shift.

We can now introduce Stochastic Variance-Reduced Policy Gradient (SVRG) for a generic policy gradient estimator $g$. Pseudo-code is provided in Algorithm 2. The overall structure is the same as Algorithm 1, but the snapshot gradient is not exact and the gradient estimate used between snapshots is corrected using importance weighting:

\[
\nabla J(\theta_t) = \nabla_N J(\tilde{\theta}) + g(\tau|\theta_t) - \omega(\tau|\theta_t, \tilde{\theta}) g(\tau|\tilde{\theta})
\]

for any $t \in \{0,\ldots,m-1\}$, where $\nabla_N J(\tilde{\theta})$ is as in Eq. (2) where $D_N$ is sampled using the snapshot policy $\pi_{\tilde{\theta}}$, is sampled from the current policy $\pi_{\theta}$, and $\omega(\tau|\theta_t, \tilde{\theta}) = p(\tau|\theta_t, \tilde{\theta}) / p(\tau|\theta_t)$ is an importance weight from $\theta_t$ to the snapshot policy $\pi_{\tilde{\theta}}$. Similarly to SVRG, we have that $\theta_0 := \tilde{\theta}$, and the update is a FG step. Our update is still fundamentally on-policy since the weighting concerns only the correction term. However, this partial “off-policyness” represents an additional source of variance. This is a well-known issue of importance sampling (e.g., Thomas et al., 2015). To mitigate it, we use mini-batches of trajectories of size $B \ll N$ to average the correction, i.e.,

\[
\nabla J(\theta_t) := v_t = \nabla_N J(\tilde{\theta}) + \frac{1}{B} \sum_{i=0}^{B-1} \left[ g(\tau_i|\theta_t) - \omega(\tau_i|\theta_t, \tilde{\theta}) g(\tau_i|\tilde{\theta}) \right].
\]

It is worth noting that the full gradient and the correction term have the same expected value:

\[
\mathbb{E}_{\tau_t \sim p(\cdot|\theta_t)} \left[ \frac{1}{B} \sum_{i=0}^{B-1} \omega(\tau_i|\theta_t, \tilde{\theta}) g(\tau_i|\tilde{\theta}) \right] = \nabla_j J(\theta_t).
\]

This property will be used to prove Lemma 3.1. The use of mini-batches is also common practice in SVRG since it can yield a performance improvement even in the supervised case (Harikandeh et al., 2015; Konečný et al., 2016). It is easy to show that the SVRG estimator has the following, desirable properties:

**Lemma 3.1.** Let $\nabla_N J(\theta)$ be an unbiased estimator of (1) and let $\theta^* \in \arg\min_{\theta} J(\theta)$. Then, the SVRG estimate

\[
\text{Algorithm 2 SVRPG}
\]

**Input:** number of epochs $S$, epoch size $m$, step size $\alpha$, batch size $N$, mini-batch size $B$, gradient estimator $g$, initial parameter $\theta_0 := \theta := \theta_0$

**for** $s = 0$ to $S - 1$

$\theta_{s+1} := \tilde{\theta} := \theta_{s}^m$

Sample $N$ trajectories $\{\tau_j\}$ from $p(\cdot|\theta_{s}^m)$

$\tilde{\mu} = \nabla_N J(\tilde{\theta})$ (see Eq. (2))

**for** $t = 0$ to $m - 1$

Sample $B$ trajectories $\{\tau_i\}$ from $p(\cdot|\theta_{s}^{t+1})$

$c_t^{s+1} = \frac{1}{\pi} \sum_{i=0}^{B-1} \left( g(\tau_i|\theta_{s}^{t+1}) - \omega(\tau_i|\theta_{s}^{t+1}, \tilde{\theta}) g(\tau_i|\tilde{\theta}) \right)$

$v_t^{s+1} = \tilde{\mu} + c_t^{s+1}$

$\theta_{s+1}^{t+1} = \theta_{s}^{t+1} + \alpha v_t^{s+1}$

**end for**

**end for**

return $\theta_A := \theta_{s+1}^m$ with $(s, t)$ picked uniformly at random from $\{[0, S - 1] \times [0, m - 1]\}$

\(^3\)See Appendix C for more details on the Gaussian policy case.

\(^4\)Note that $g$ can be any unbiased estimator, with or without baseline. The unbiasedness is required for theoretical results (e.g., Appendix A).

\(^5\)The reader can refer to Appendix A for off-policy gradients and variants of REINFORCE and G(PO)MDP.
in (4) is unbiased
\[ \mathbb{E} [\nabla J(\theta)] = \nabla J(\theta). \] (5)
and regardless of the mini-batch size \( B \).

\[ \mathbb{V} \text{ar} [\nabla J(\theta^*)] = \mathbb{V} \text{ar} [\hat{\nabla} N J(\theta^*)]. \] (6)

Previous results hold for both REINFORCE and G(PO)MDP. In particular, the latter result suggests that an SVRG-like algorithm using \( \nabla J(\theta) \) can achieve faster convergence, by performing much more parameter updates with the same data without introducing additional variance (at least asymptotically). Note that the randomized return value of Algorithm 2 does not affect online learning at all, but will be used as a theoretical tool in the next section.

**4. Convergence Guarantees of SVRPG**

In this section, we state the convergence guarantees for SVRPG with REINFORCE or G(PO)MDP gradient estimator. We mainly leverage on the recent analysis of non-concave SVRG (Reddi et al., 2016a; Allen-Zhu & Hazan, 2016). Each of the three challenges presented at the beginning of Section 3 can potentially prevent convergence, so we need additional assumptions. In Appendix C we show how Gaussian policies satisfy these assumptions.

1) Non-concavity. A common assumption, in this case, is to assume the objective function to be \( L \)-smooth. However, in RL we can consider the following assumption which is sufficient for the \( L \)-smoothness of the objective (see Lemma B.2).

**Assumption 4.1** (On policy derivatives). For each state-action pair \((s, a)\), any value of \( \theta \), and all parameter components \( i, j \) there exist constants \( 0 \leq G, F < \infty \) such that:
\[ |\nabla_{\theta_i} \log \pi_{\theta}(a|s)| \leq G, \quad \left| \frac{\partial^2}{\partial_{\theta_i} \partial_{\theta_j}} \log \pi_{\theta}(a|s) \right| \leq F. \]

2) FG Approximation. Since we cannot compute an exact full gradient, we require the variance of the estimator to be bounded. This assumption is similar in spirit to the one in (Harikandeh et al., 2015).

**Assumption 4.2** (On the variance of the gradient estimator). There is a constant \( V < \infty \) such that, for any policy \( \pi_{\theta} \):
\[ \mathbb{V} \text{ar} [g(\cdot|\theta)] \leq V. \]

3) Non-stationarity. Similarly to what is done in SL (Cortes et al., 2010), we require the variance of the importance weight to be bounded.

---

\( ^6 \) For any vector \( \mathbf{x} \), we use \( \mathbb{V} \text{ar}[\mathbf{x}] \) to denote the trace of the covariance matrix, i.e., \( \text{Tr}(\mathbb{E} [(\mathbf{x} - \mathbb{E} [\mathbf{x}]) (\mathbf{x} - \mathbb{E} [\mathbf{x}])^T]) \).
a variant of SVRPG exploiting a variance-reduction technique for importance weights. Despite lacking theoretical guarantees, we will show in Section 7 that this method can outperform the baseline SVRPG (Algorithm 2).

5.1. Full Gradient Update

As noted in Section 3, the update performed at the beginning of each epoch is equivalent to a full-gradient update. In our setting, where collecting samples is particularly expensive, the $B$ trajectories collected using the snapshot trajectory $\pi_\theta$ feels like a waste of data (the term $\sum_i g(\tau_i) - \omega(\tau_i)g(\tau_i) = 0$ since $\theta_0 = \bar{\theta}$). In practice, we just perform an approximate full gradient update using the $N$ trajectories sampled to compute $\nabla_N J(\bar{\theta})$, i.e.,

$$\theta_{t+1} = \bar{\theta} + \alpha \nabla_N J(\bar{\theta})$$

$$\theta_{t+1} = \theta_t + \alpha \nabla J(\theta_{t+1})$$

for $t = 1, \ldots, m - 1$.

In the following, we will always use this practical variant.

5.2. Meta-Parameter Selection

The step size $\alpha$ is crucial to balance variance reduction and efficiency, while the epoch length $m$ influences the variance introduced by the importance weights. Low values of $m$ are associated with small variance but increase the frequency of snapshot points (which means many FG computations). High values of $m$ may move policy $\pi_\theta$ far away from the snapshot policy $\pi_\theta$, causing large variance in the importance weights. We will jointly set the two meta-parameters.

Adaptive step size. A standard way to deal with noisy gradients is to use adaptive strategies to compute the step size. ADAdaptive Moment estimation (ADAM) (Kingma & Ba, 2014) stabilizes the parameter update by computing learning rates for each parameter based on an incremental estimate of the gradient variance. Due to this feature, we would like to incorporate ADAM in the structure of the SVRPG update. Recall that SVRPG performs two different updates of the parameters $\theta$: I) FG update in the snapshot; II) corrected gradient update in the sub-iterations. Given this structure, we suggest using two separate ADAM estimators:

$$\theta_{t+1} = \bar{\theta} + \alpha_{FG} \nabla_N J(\bar{\theta})$$

$$\theta_{t+1} = \theta_t + \alpha_{SI,t} \nabla J(\theta_{t+1})$$

where $\alpha_{FG}$ is associated with the snapshot and $\alpha_{SI,t}$ with the sub-iterations (see Appendix D for details). By doing so, we decouple the contribution of the variance due to the approximate FG from the one introduced by the sub-iterations. Note that these two terms have different orders of magnitude since are estimated with a different number of trajectories ($B \ll N$) and the estimator in the snapshot does not require importance weights. The use of two ADAM estimators allows to capture and exploit this property.

Adaptive epoch length. It is easy to imagine that a predefined schedule (e.g., $m$ fixed in advance or changed with a policy-independent process) may poorly perform due to the high variability of the updates. In particular, given a fixed number of sub-iterations $m$, the variance of the updates in the sub-iterations depends on the snapshot policy and the sampled trajectories. Since the ADAM estimate partly captures such variability, we propose to take a new snapshot (i.e., interrupt the sub-iterations) whenever the step size $\alpha_{SI}$ proposed by ADAM for the sub-iterations is smaller than the one for the FG (i.e., $\alpha_{FG}$). If the latter condition is verified, it amounts to say that the noise in the corrected gradient has overcome the information of the FG. Formally, the stopping condition is as follows

$$\frac{\alpha_{FG}}{N} > \frac{\alpha_{SI}}{B}$$

where we have introduced $N$ and $B$ to take into account the trajectory efficiency (i.e., weighted advantage). The less the number of trajectories used to update the policy, the better. Including the batch sizes in the stopping condition allows us to optimize the trade-off between the quality of the updates and the cost of performing them.

5.3. Normalized Importance Sampling

As mentioned in Section 5.2, importance weights are an additional source of variance. A standard way to cope with this issue is self-normalization (e.g., Precup, 2000; Owen, 2013). This technique can reduce the variance of the importance weights at the cost of introducing some bias (Owen, 2013, Chapter 9). Whether the trade-off is advantageous depends on the specific task. Introducing self-normalization in the context of our algorithm, we switch from Eq. (4) to:

$$\nabla J(\theta_t) = \nabla_N J(\bar{\theta}) + \frac{1}{B} \sum_{i=0}^{B-1} [g(\tau_i|\theta_t)] - \frac{1}{\Omega} \sum_{i=0}^{B-1} \left[ \omega(\tau_i|\theta_t, \bar{\theta})g(\tau_i|\bar{\theta}) \right].$$

where $\Omega = \sum_{i=0}^{B-1} \omega(\tau_i|\theta_t, \bar{\theta})$. In Section 7 we show that self-normalization can provide a performance improvement.

6. Related Work

Despite the considerable interest received in SL, variance-reduced gradient approaches have not attracted the RL community. As far as we know, there are just two applications of SVRG in RL. The first approach (Du et al., 2017) aims to exploit SVRG for policy evaluation. The policy evaluation problem is more straightforward than the one faced in this paper (control problem). In particular, since the goal is to evaluate just the performance of a predefined policy, the optimization problem is stationary. The setting considered in the paper is the one of policy evaluation by minimizing the
empirical mean squared projected Bellman error (MSPBE) with a linear approximation of the value function. Du et al. (2017) shown that this problem can be equivalently reformulated as a convex-concave saddle-point problem that is characterized by a finite-sum structure. This problem can be solved using a variant of SVRG (Palaniappan & Bach, 2016) for which convergence guarantees have been provided. The second approach (Xu et al., 2017) uses SVRG as a practical method to solve the optimization problem faced by Trust Region Policy Optimization (TRPO) at each iteration. This is just a direct application of SVRG to a problem having finite-sum structure since no specific structure of the RL problem is exploited. It is worth to mention that, for practical reasons, the authors proposed to use a Newton conjugate gradient method with SVRG.

In the recent past, there has been a surge of studies investigating variance reduction techniques for policy gradient methods. The specific structure of the policy gradient allows incorporating a baseline (i.e., a function $b: S \times A \to \mathbb{R}$) without affecting the unbiasedness of the gradient (e.g., Williams, 1992; Weaver & Tao, 2001; Peters & Schaal, 2008b; Thomas & Brunskill, 2017; Wu et al., 2018). Although the baseline can be arbitrarily selected, literature often refers to the optimal baseline as the one minimizing the variance of the estimate. Nevertheless, even the baseline needs to be estimated from data. This fact may partially reduce its effectiveness by introducing variance. Even if these approaches share the same goal as SVRG, they are substantially different. In particular, the proposed SVRPG does not make explicit use of the structure of the policy gradient framework, and it is independent of the underlying gradient estimate (i.e., with or without baseline). This suggests that would be possible to integrate an ad-hoc SVRPG baseline to further reduce the variance of the estimate. Since this paper is about the applicability of SVRG technique to RL, we consider this topic as future work. Additionally, the experiments show that SVRPG has an advantage over G(PO)MPD even when the baseline is used (see the half-cheetah domain in Section 7).

Concerning importance weighting techniques, RL has made extensive use of them for off-policy problems (e.g., Precup, 2000; Thomas et al., 2015). However, as mentioned before, SVRPG cannot be compared to such methods since it is in all respects an on-policy algorithm. Here, importance weighting is just a statistical tool used to preserve the unbiasedness of the corrected gradient.

7. Experiments

In this section, we evaluate the performance of SVRPG and compare it with policy gradient (PG) on well known continuous RL tasks: Cart-pole balancing and Swimmer (e.g., Duan et al., 2016). We consider G(PO)MDP since it has a smaller variance than REINFORCE. For our algorithm, we use a batch size $N = 100$, a mini-batch size $B = 10$, and the jointly adaptive step size $\alpha$ and epoch length $m$ proposed in Section 5.2. Since the aim of this comparison is to show the improvement that SVRG-flavored variance reduction brings to SG in the policy gradient framework, we set the batch size of the baseline policy gradient algorithm to $B$. In this sense, we measure the improvement yielded by computing snapshot gradients and using them to adjust parameter updates. Since we evaluate on-line performance over the number of sampled trajectories, the cost of computing such snapshot gradients is automatically taken into consideration. To make the comparison fair, we also use Adam in the baseline PG algorithm, which we will denote simply as G(PO)MDP in the following. In all the experiments, we use deep Gaussian policies with adaptive standard deviation (details on network architecture in Appendix E). Each experiment is run 10 times with a random policy initialization and seed, but this initialization is shared among the algorithms under comparison. The length of the experiment, i.e., the total number of trajectories, is fixed for each task. Performance is evaluated by using test-trajectories on a subset of the policies considered during the learning process. We provide average performance with 90% bootstrap confidence intervals. Task implementations are from the rllab library (Duan et al., 2016), on which our agents are also based. More details on meta-parameters and exhaustive task descriptions are provided in Appendix E.

Figure 1a compares SVRPG with G(PO)MDP on a continuous variant of the classical Cart-pole task, which is a 2D balancing task. Despite using more trajectories on average for each parameter update, our algorithm shows faster convergence, which can be ascribed to the better quality of updates due to variance reduction.

The Swimmer task is a 3D continuous-control locomotion task. This task is more difficult than cart-pole. In particular, the longer horizon and the more complex dynamics can have a dangerous impact on the variance of importance weights. In this case, the self-normalization technique proposed in Section 5.3 brings an improvement (even if not statistically significant), as shown in Figure 1b. Figure 1c shows self-normalized SVRPG against G(PO)MDP. Our algorithm outperforms G(PO)MDP for almost the entire learning process. Also here, we note an increase of speed in early iterations, and, toward the end of the learning process, the improvement becomes statistically significant.

Preliminary results on actor-critic. Another variance-reduction technique in policy gradient consists of using baselines or critics. This tool is orthogonal to the methods described in this paper, and the theoretical results of Section 4 are general in this sense. In the experiments de-

7Code available at github.com/Dam930/rllab.
scribed so far, we compared against the so-called actor-only G(PO)MDP, i.e., without the baseline. To move towards a more general understanding of the variance issue in policy gradient, we also test SVRPG in an actor-critic scenario. To do so, we consider the more challenging MuJoCo (Todorov et al., 2012) Half-cheetah task, a 3D locomotion task that has a larger state-action space than Swimmer. Figure 1d compares self-normalized SVRPG and G(PO)MDP on Half-cheetah, using the critic suggested in (Duan et al., 2016) for both algorithms. Results are promising, showing that a combination of the baseline usage and SVRG-like variance reduction can yield an improvement that the two techniques alone are not able to achieve. Moreover, SVRPG presents a noticeably lower variance. The performance of actor-critic G(PO)MDP\(^8\) on Half-Cheetah is coherent with the one reported in (Duan et al., 2016). Other results are not comparable since we did not use the critic.

8. Conclusion

In this paper, we introduced SVRPG, a variant of SVRG designed explicitly for RL problems. The control problem considered in the paper has a series of difficulties that are not common in SL. Among them, non-concavity and approximate estimates of the FG have been analysed independently in SL (e.g., Allen-Zhu & Hazan, 2016; Reddi et al., 2016a; Harikandeh et al., 2015) but never combined. Nevertheless, the main issue in RL is the non-stationarity of the sampling process since the distribution underlying the objective function is policy-dependent. We have shown that by exploiting importance weighting techniques, it is possible to overcome this issue and preserve the unbiasedness of the corrected gradient. We have additionally shown that, under mild assumptions that are often verified in RL applications, it is possible to derive convergence guarantees for SVRPG. Finally, we have empirically shown that practical variants of the theoretical SVRPG version can outperform classical actor-only approaches on benchmark tasks. Preliminary results support the effectiveness of SVRPG also with a commonly used baseline for the policy gradient. Despite that, we believe that it will be possible to derive a baseline designed explicitly for SVRPG to exploit the RL structure and the SVRG idea jointly. Another possible improvement would be to employ the natural gradient (Kakade, 2002) to better control the effects of parameter updates on the variance of importance weights. Future work should also focus on making batch sizes \(N\) and \(B\) adaptive, as suggested in (Papini et al., 2017).
Acknowledgments

This research was supported in part by French Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council and French National Research Agency (ANR) under project ExTra-Learn (n.ANR-14-CE24-0010-01).

References


Owen, Art B. Monte Carlo theory, methods and examples. 2013.


