Geometric Value Iteration: Dynamic Error-Aware KL Regularization for Reinforcement Learning

Toshinori Kitamura  
KITAMURA.TOSHINORI.KT6@IS.NAIST.JP
Lingwei Zhu  
zhu.lingwei.zj5@IS.NAIST.JP
Takamitsu Matsubara  
TAKAM-M@IS.NAIST.JP
Nara Institute of Science and Technology, Nara, JAPAN

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

The recent boom in the literature on entropy-regularized reinforcement learning (RL) approaches reveals that Kullback-Leibler (KL) regularization brings advantages to RL algorithms by canceling out errors under mild assumptions. However, existing analyses focus on fixed regularization with a constant weighting coefficient and do not consider cases where the coefficient is allowed to change dynamically. In this paper, we study the dynamic coefficient scheme and present the first asymptotic error bound. Based on the dynamic coefficient error bound, we propose an effective scheme to tune the coefficient according to the magnitude of error in favor of more robust learning. Complementing this development, we propose a novel algorithm, Geometric Value Iteration (GVI), that features a dynamic error-aware KL coefficient design with the aim of mitigating the impact of errors on performance. Our experiments demonstrate that GVI can effectively exploit the trade-off between learning speed and robustness over uniform averaging of a constant KL coefficient. The combination of GVI and deep networks shows stable learning behavior even in the absence of a target network, where algorithms with a constant KL coefficient would greatly oscillate or even fail to converge.

Keywords: Geometric Policy Interpolation; Error-Awareness; KL Regularization; Reinforcement Learning

1. Introduction

The recently impressive successes of reinforcement learning (RL) rely heavily on the use of nonlinear function approximators such as deep networks (Mnih et al., 2015; Silver et al., 2017). However, the power of nonlinear approximators comes at a cost that approximation or estimation errors can easily go uncontrolled due to stochastic approximation using noisy samples (Fu et al., 2019), leading to performance oscillation or even divergent learning (Lillicrap et al., 2015; Fujimoto et al., 2018). While error propagation has been studied in detail in the literature of approximate dynamic programming (ADP) methods (Munos and Szepesvári, 2008; Scherrer et al., 2015), little is understood in the case of nonlinear approximation, such as whether the analyses in ADP still hold true. In practice, several empirical tricks like target networks or asynchronous updates need to be used to ensure stability and convergence for learning with deep networks (Mnih et al., 2015; Haarnoja et al., 2018).

The recent boom in the literature on entropy-regularized RL highlights the use of Kullback-Leibler (KL) divergence as a regularization term in the reward (Azar et al., 2012; Kozuno et al., 2019; Vieillard et al., 2020a). It is known that by adding KL regularization, errors are grouped in the sense that they are accumulated as a summation (more details in Section 2). In standard $L_p$ norm error propagation analysis, this summation is within the norm, as compared to the summation-over-norm of the standard ADP results (Bertsekas and Tsitsiklis, 1996; Munos and Szepesvári, 2008). Under mild assumptions such as the sequence of errors having martingale difference, the summation of errors asymptotically cancels out. This brings a great advantage to deep RL, where properly addressing errors is paramount (Fu et al., 2019; Fujimoto et al., 2018). However, there is a trade-off between learning speed and robustness in play since the policies change less between iterations with KL regularization. By setting the KL regularization coefficient as a constant, we lose the ability to dynamically trade-off speed and robustness, and hence the resultant algorithms might not be suitable for robustness-critical problems. In practice, wild performance oscillation can indeed be observed (Nachum et al., 2018), since summation is still sensitive to outliers and the errors at the early stage of learning are typically large.

In this paper, we propose dynamically adjusting the KL regularization coefficient according to the error made at each iteration, with the motivation being that for iterations with large error, large KL regularization weight should be imposed to prevent the agent from going in the wrong update direction. We prove the resulting error propagation bound has the form of norm-over-weighted-summation (Theorem 2), which has the potential to more effectively improve the trade-off between learning speed and robustness than uniform averaging.

The rest of the paper is organized as follows. We introduce the notations used and review existing constant KL coefficient RL algorithms in Section 2. In Section 3, we study ADP with a dynamic KL coefficient. Specifically, we discuss our novel design of the KL coefficient, which is based on the maximum iteration-wise error for weighting the effect of regularization. Based on the KL coefficient design, in Section 4 we present a practical RL algorithm, Geometric Value Iteration (GVI). We evaluate GVI on simple mazes and a set of classic control tasks in Section 5. Our experiments show that GVI can converge faster and more stably and that, moreover, GVI with a deep neural network demonstrates significantly stabilized learning compared to constant regularization, even without target networks. Related works and a discussion are given in Section 6. Section 7 presents our conclusions.

2. Background and Notations

We consider a discounted Markov Decision Process (MDP) defined by a tuple $\{S, A, P, r, d_0, \gamma\}$, where $S$ is the finite state space, $A$ is the finite set of actions, $P \in \Delta_{S \times A}^S$ is the transition kernel (writing $\Delta_X$ as the probability simplex over the set $X$, and $X^Y$ is the set of applications from $X$ to $Y$), $r \in \mathbb{R}^{S \times A}$ is the reward function bounded by $r_{max}$, $d_0 \in \Delta_S$ is the distribution of the initial state, and $\gamma \in (0, 1)$ is the discount factor. A policy $\pi \in \Delta_S$ maps states to a distribution over actions, and we write the expectation over trajectories induced by $\pi$ and $d_0$ as $\mathbb{E}_\pi$, where we omit the notation $d_0$ for simplicity. For a policy $\pi$, the state-action
value function is defined as \( q_\pi(s,a) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(S_t, A_t) \big| S_0 = s, A_0 = a \right] \), and the (un-normalized) discounted visitation frequency is defined as \( d_\pi(s) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t P(S_t = s) \right] \).

Following Vieillard et al. (2020a), we define a component-wise dot product \((f_1, f_2) = (\sum_a f_1(s,a) f_2(s,a))_s \in \mathbb{R}^S\) for \(f_1, f_2 \in \mathbb{R}^{S \times A}\), which is useful for expectation calculations. We define \( P_v = (\sum_v P(s'|s,a)v(s'))_{s,a} \in \mathbb{R}^{S \times A}\) for \(v \in \mathbb{R}^S\). We also define a policy-induced transition kernel \( P_\pi \) as \( P_\pi q = P(\pi, q) \). We write the Bellman evaluation operator \( T_\pi q = r + \gamma P_\pi q \) and its unique fixed point as \( q_\pi \). An optimal policy satisfies \( \pi_* = \arg\max_{\pi \in \Delta^S_A} q_\pi \), and \( q_* = q_{\pi_*} \). We denote the set of greedy policies w.r.t. \( q \in \mathbb{R}^{S \times A} \) as \( \mathcal{G}(q) = \arg\max_{\pi \in \Delta^S_A} \langle q, \pi \rangle \). When scalar functions are applied to vectors, their applications should be understood in a point-wise fashion. KL divergence and Shannon entropy are the two most widely used entropy terms for regularization. We express KL divergence as \( \text{KL}(\pi_1 \| \pi_2) = \langle \pi_1, \ln \pi_1 - \ln \pi_2 \rangle \in \mathbb{R}^S \) and Shannon entropy (or simply entropy) as \( H(\pi) = \langle -\pi, \ln \pi \rangle \in \mathbb{R}^S \).

**Mirror Descent Value Iteration** Vieillard et al. (2020a) provides a generalized framework for KL-regularized ADP schemes. The framework, termed Mirror Descent Policy Iteration (MD-PI), is given in Eq. (1), where the equation sign indicates the component-wise update of a vector. While MD-PI can also consider the popular Shannon entropy regularization (Haarnoja et al., 2017), Shannon entropy does not provide an advantage in the theoretical error propagation analysis of MD-PI (Vieillard et al., 2020a). Accordingly, in this paper, we focus only on the KL regularization.

\[
\begin{aligned}
\text{MD-PI} \quad \begin{cases}
\pi_{k+1}^\lambda &= \mathcal{G}^\lambda_{\pi_k}(q_k) \\
q_{k+1} &= (T^\lambda_{\pi_{k+1}} \pi_k)^m q_k + \epsilon_{k+1}
\end{cases}
\end{aligned}
\]

In Eq. (1), we start from a uniform policy \( \pi_0 \) and evaluate the next policy by applying the greedy operator \( \mathcal{G}^\lambda_{\mu}(q) = \arg\max_{\pi \in \Delta^S_A} \langle \pi, q \rangle - \lambda \text{KL}(\pi \| \mu) \) with an arbitrary baseline policy \( \mu \in \Delta^S_A \). Usually, \( \mu \) is chosen as the previous policy. With the obtained regularized greedy policy, we evaluate its action value function \( q_{k+1} \) by applying \( m \) times the regularized Bellman operator \( T^\lambda_{\pi_{k+1}} \| q = r + \gamma P(\pi, q) - \lambda \text{KL}(\pi \| \mu) \). Setting \( m = 1, \infty \) corresponds to value iteration and policy iteration schemes, respectively. Setting \( m \) to any other value implies the use of approximate modified policy iteration (Puterman and Shin, 1978; Scherrer et al., 2015). We call Eq. (1) with \( m = 1 \) the Mirror Descent Value Iteration (MD-VI). The error term \( \epsilon_{k+1} \) is a vector of the same shape as the action value function, and it is typically assumed that the greedy step is free of error (Vieillard et al., 2020a).

By the Fenchel conjugacy (Boyd and Vandenberghe, 2004), the greedy policy \( \pi_{k+1} \) can be analytically obtained as \( \mathcal{G}^\lambda_{\mu}(q) \propto \mu \exp \left( \frac{q}{\lambda} \right) \) (Geist et al., 2019). By choosing \( \mu = \pi_k \), a direction induction shows that the MD-PI policy \( \pi_{k+1} \) averages all previous Q-values as \( \pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda} \times \cdots \times \exp \frac{q_1}{\lambda} \sum_{j=0}^k q_j \). Since the errors are additive, \( \pi_{k+1} \) also averages errors from previous iterations. Indeed, the following theorem formally shows that the finite-time bound of MD-VI depends on the norm of the average of the accumulated errors (the extension to \( m > 1 \) remaining an open question).
Theorem 1 (Vieillard et al. (2020a)) Define the maximum value of $\|q_k\|_\infty$ as $q_{\max}$.

The $\ell_\infty$-bound of MD-VI is

$$\|q_* - q_{\pi_{k+1}}\|_\infty \leq \frac{2}{(1 - \gamma)} \frac{1}{k} \left( \left\| \sum_{j=1}^{k} \epsilon_j \right\|_\infty + 2q_{\max} + \lambda \gamma \ln |A| \right).$$

In Theorem 1, the optimality gap $\|q_* - q_{\pi_{k+1}}\|_\infty$ is expressed in terms of errors $\frac{1}{k} \sum_{j=1}^{k} \epsilon_j$, which are averaged with respect to the uniform distribution. This corresponds to having a constant coefficient-KL regularization throughout learning (i.e., fixing $\lambda$). Under mild assumptions, such as the sequence of errors having martingale difference under the natural filtration (Azar et al., 2012), the summation of errors asymptotically cancels out. However, the asymptotic cancelation of errors happens only under specific conditions. When the conditions are not satisfied, having a constant coefficient assumes the errors contribute equally (i.e., $\frac{1}{k}$) to the gap $\|q_* - q_{\pi_{k+1}}\|_\infty$, which is often not the case, since in the early stages of learning the errors are typically large and require more attention.

Our motivation comes from the intuition of weighting down large errors using large regularization coefficients $\lambda_k$, and thus the weighted average of errors $\frac{1}{\sum_{j=1}^{k} 1/\lambda_j} \left\| \sum_{j=1}^{k} \epsilon_j / \lambda_j \right\|_\infty$ could be much smaller than that of uniform averaging $\frac{1}{k} \left\| \sum_{j=1}^{k} \epsilon_j \right\|_\infty$. This corresponds to setting a different KL coefficient for each iteration. Intuitively, different coefficients allow for more robust convergence and potentially faster convergence since the magnitude of $\frac{1}{\sum_{j=1}^{k} 1/\lambda_j} (\epsilon_j / \lambda_j)$ could be much smaller than $\frac{1}{k} \epsilon_j$ if we are allowed to specify the coefficient $\lambda_j$. This motivation prompts the use of a dynamic error-aware KL coefficient design that is detailed in Section 3.

3. Dynamic Error-Aware KL Regularization

While MD-VI is generally robust against zero-mean errors because the summation of errors asymptotically cancels out, in some situations the errors fail to cancel each other out and result in bad performance of MD-VI. As a concrete example, consider the following errors induced every $K$ step:

$$\begin{cases} 
\epsilon_k \sim \text{unif}(0, K) & \text{if } k = K, 2K, \ldots \\
\epsilon_k = 0 & \text{otherwise}
\end{cases}$$

This artificial example can be likened to a two-state MDP case (Figure 2) where the agent starting from state 1 continues to loop onto itself with zero cost and probability $\frac{K-1}{K}$, and with probability $\frac{1}{K}$ the agent goes to state 2 with cost $\text{unif}(0, K)$ and then back again to state 1.

Figure 1 illustrates the optimality gap of MD-VI under randomly generated $5 \times 5$ mazes and errors of Eq. (3) with $K = 100$ (see the environment’s details in Appendix D). In this simple setup, trials with small regularization coefficients (blue line) fail to converge due to performance oscillation brought by the error in Eq. (3), while larger regularization (yellow and green lines) achieves convergence to the optimal policy but at a much slower rate. A suitable strategy in this example is obviously an error-aware regularization strategy: being conservative only when the errors are present, and greedy otherwise.
To overcome the limitations of the constant-weight regularization scheme, we study the following regularized PI scheme with a dynamic KL coefficient $\lambda$:

$$
\begin{align*}
\pi_{k+1} &= G^{\lambda_k}_{\pi_k}(q_k) \\
q_{k+1} &= (T^{\lambda_k}_{\pi_{k+1}|\pi_k})^m q_k + \epsilon_{k+1}.
\end{align*}
$$

(4)

We derive the error-aware regularization bound for the policy iteration case. The following theorem provides a bound on the optimality gap of Eq. (4).

**Theorem 2** Define $\eta_k = 1/\lambda_k$ and $Z_k = \sum_{j=0}^{k} \eta_j$. The $\ell_\infty$-bound of Eq. (4) with $m = 1$ is

$$
\|q_* - q_{\pi_{k+1}}\|_\infty \leq \frac{2}{(1 - \gamma) Z_k} \left( \left\| \sum_{j=1}^{k} \eta_j \epsilon_j \right\|_\infty + (\eta_{k+1} + \eta_0 + \sum_{j=0}^{k} |\eta_{j+1} - \eta_j|) q_{\max} + \gamma \ln |A| \right).
$$

(5)

**Proof** See Appendix A for the proof.

Note that this bound generalizes the bound of MD-VI, since Eq. (5) matches Eq. (2) when $\eta = 1/\lambda$. As with MD-VI, this bound features a linear dependency of errors on the horizon $1/(1 - \gamma)$. The main difference is the **weighted average** of the errors instead of the **uniform average** for MD-VI. This weighted average error term intuitively motivates the design of regularization coefficients $\eta_k = 1/\lambda_k$.

First, minimizing the optimality gap implies minimizing $\|\frac{1}{Z_k} \sum_{j=1}^{k} \eta_j \epsilon_j\|_\infty$, which is the norm of the weighted arithmetic mean of errors. Because $\epsilon_j$ is a random variable for all $j$, with $\eta_j' = \frac{\eta_j}{Z_k}$, the mean and the variance of the weighted arithmetic mean are given by $\sum_{j=1}^{k} \eta_j' \mathbb{E}[\epsilon_j]$ and $\sum_{j=1}^{k} \eta_j'^2 \text{Var}[\epsilon_j]$, respectively. This in turn suggests that $\eta_j$ should be inversely scaled according to the magnitude of $\epsilon_j$ to restrict potentially erroneous updates.
where errors have huge means or variances. By recalling $\epsilon_j$ is a vector, we scale $\eta_j$ according to the infinity norm as $\eta_j = \frac{1}{\alpha_1 \|\epsilon_j\|_\infty}$, where $\alpha_1$ is used for uniformly scaling all of the coefficients. Note that $\alpha_1$ does not appear in the error-dependent term since it appears in both numerator $\eta_j$ and denominator $Z_k$.

In deep RL, hyperparameters are typically and gradually decayed instead of changed abruptly. This highlights the importance of stability in learning with neural networks, which we address here. Given the above design choice, we impose an additional constraint that no huge increase from $\eta_k$ to $\eta_{k+1}$ is allowed: such an increase during learning can be measured by $\frac{1}{Z_k} \sum_{j=0}^{k} |\eta_{j+1} - \eta_j|$, which appears in the second term of the error bound Eq. (5). We do not allow the term to diverge by restricting $\eta_{j+1} > 2\eta_j$, which makes $\frac{1}{Z_k} \sum_{j=0}^{k-1} |\eta_{j+1} - \eta_j|$ larger than 1. To this end, we gradually decay the regularization coefficient by introducing another hyperparameter $\alpha_2$, such that $\lambda_k = \alpha_2 \lambda_{k-1}$ with $\alpha_2 \in (0, 1)$ generally close to one.

The above-mentioned design choices can be summarized as the following dynamic KL coefficient design:

$$\lambda_k = \max(\alpha_1 \|\epsilon_k\|_\infty, \alpha_2 \lambda_{k-1}),$$

where $\alpha_1 \in \mathbb{R}^+$ and $\alpha_2 \in (0, 1)$.

4. Geometric Value Iteration

In this section we propose a novel algorithm based on the dynamic KL regularization coefficient design of the previous section. While it is straightforward to incorporate it in the general MD-VI scheme of Eq. (4), a crucial subtlety stands in the way of achieving better performance: we know $\pi_{k+1} \propto \exp(\sum_{j=1}^{k} q_j)$ from Section 2, which requires remembering all previous value functions. In practice, approximation such as information projection would have to be used (Vieillard et al., 2020d), which brings errors to the policy update step.

Leveraging the very recent idea of implicit KL regularization (Vieillard et al., 2020c), it is possible to circumvent the need for remembering all previous values in MD-VI by augmenting the reward with a log-policy term, whose formulation is given in Eq. (7). The reward function is augmented by the term $\ln \pi_{k+1}$ weighted by the KL coefficient $\lambda$:

$$\begin{align*}
\pi_{k+1} &= \arg\max_{\pi \in \Delta_{\mathcal{A}}} \langle \pi, q_k \rangle - \lambda \mathcal{H}(\pi) \\
q_{k+1} &= \lambda \ln \pi_{k+1} + r + \gamma P \langle \pi_{k+1}, q_k - \lambda \ln \pi_{k+1} \rangle.
\end{align*}$$

Eq. (7) corresponds to implicitly performing KL regularization, and hence there is no need for remembering previous values, that is, computing the term $\ln \pi_{k+1}$ suffices.

While Eq. (7) provides an easy-to-use scheme for our dynamic KL coefficient by replacing $\lambda$ with $\lambda_k$, the term $\lambda_k \ln \pi_k$ could cause numerical issues when $\lambda_k$ has a huge value. For numerical stability, we propose further transforming Eq. (7) as follows:

$$\begin{align*}
\pi_{k+1} &= \arg\max_{\pi \in \Delta_{\mathcal{A}}} \langle \pi, q_k \rangle + \mathcal{H}(\pi) \\
q_{k+1} &= \ln \pi_{k+1} + \frac{r}{\pi_{k+1}} + \frac{\lambda_k}{\pi_{k+1}} \gamma P \langle \pi_{k+1}, q_k - \ln \pi_{k+1} \rangle.
\end{align*}$$

Additional clipping might also be necessary to restrict the magnitude of $\ln \pi_{k+1}$. We can show that the scheme of Eq. (8) is equivalent to the formulation of Eq. (4), which we formally state below.
Theorem 3 For any $k \geq 0$, by defining $q_k' = \lambda_{k+1} (q_k - \ln \pi_k)$, we have

$$\pi_{k+1} = \arg\max_{\pi \in \Delta_s^A} \langle \pi, q_k' \rangle - \lambda_k KL(\pi\|\pi_k)$$

$$q_{k+1}' = r + \gamma P \langle \pi_{k+1}, q_k' \rangle - \lambda_k KL(\pi_{k+1}\|\pi_k)$$

(9)

Proof See Appendix B for the proof.

By dynamically adjusting the KL coefficient, Eq. (8) mitigates issues brought by various sources of error and improves learning stability. One more problem remains for making Eq. (8) practically applicable. In tuning the KL coefficient Eq. (6), the magnitude information of $\|\epsilon_{k+1}\|_\infty$ is typically unavailable. Taking inspiration from a very recent work (Vieillard et al., 2020b), we approximately compute this error by moving average TD-error from batches. Hence, we approximate $|\epsilon_{k+1}|_\infty$ by the maximum absolute TD error $|\epsilon_{TD,i}|_\infty$, where $i$ indicates the $i$th batch. In summary, our Geometric Value Iteration (GVI) iterates as follows:

$$\text{GVI} \left\{ \begin{array}{l}
\lambda_{k+1} = \max(\alpha_1 \|\epsilon_{TD,k}\|_\infty, \alpha_2 \lambda_k) \\
\pi_{k+1} = \arg\max_{\pi \in \Delta_s^A} \langle \pi, q_k \rangle + H(\pi) \\
q_{k+1} = \ln \pi_{k+1} + \frac{\lambda}{\lambda_{k+1}} P \langle \pi_{k+1}, q_{k} - \ln \pi_{k+1} \rangle
\end{array} \right.$$

(10)

The name Geometric comes from the fact that GVI mixes two policies by weighted geometric mean as $\pi_{k+1} = \frac{q_{k+1}^\lambda}{q_{k+1}^\lambda} (\pi_k)^{1-\zeta_k} \left( \frac{q_{k+1}^\lambda}{q_k^\lambda} \right)^{\zeta_k}$, where $\lambda/\zeta_k = \lambda_k$.

We now present the implementation of Eq. (10) using deep networks, or Deep GVI (DGVI). Suppose $Q$-values are estimated by an online $Q$ network parameterized by weight vector $\theta$ and the transition data are stored in a FIFO replay buffer $B$. DGVI minimizes the following loss function:

$$L_\theta = \mathbb{E}_{(s,a,r,s') \sim B} \left[ (q_\theta(s, a) - y(s, a))^2 \right],$$

(11)

where $y(s, a) = \ln \pi(a|s) + \frac{r(s, a)}{\lambda'} + \frac{\lambda}{\lambda'} \gamma \mathbb{E}_{a' \sim \pi(|s')} [q_\theta(s', a') - \ln \pi(a'|s')]$,

(12)

where $\bar{\theta}$ indicates the weight vector of the target network and $\pi \propto \exp(q_\theta)$ is the greedy policy. $\lambda$ and $\lambda'$ are the previous and current KL coefficients, respectively. The use of slowly updated target network $q_\theta$ in the current $y(s, a)$ is conventional for stability purposes. The parameters $\theta$ are either infrequently copied from $\theta$ or obtained by Polyak averaging $\theta$. While target networks could be used to further enhance the performance, in our experiments we explicitly remove target networks to highlight the error-robustness of DGVI.

Taking inspiration from (Vieillard et al., 2020b), we use the moving average of maximum batch TD errors to approximate the maximum error based on Eq. (8):

$$\lambda' \leftarrow (1 - \nu)\lambda' + \nu \max(\alpha_1 \|\epsilon_{TD}\|_\infty, \alpha_2 \lambda)$$

$$\lambda \leftarrow (1 - \nu_{\text{slow}})\lambda + \nu_{\text{slow}}\lambda'$$

(13)

where $\|\epsilon_{TD}\|_\infty$ is the maximum absolute TD error in a batch, and $\nu$ and $\nu_{\text{slow}}$ are learning rates for $\lambda$ and $\lambda'$, respectively. We summarize the algorithm of DGVI in Algorithm 1.
Algorithm 1 Deep Geometric Value Iteration

1: Initialize $\theta$, $\lambda$ and $\lambda'$
2: for each iteration do
3:      Collect transitions and add them to $B$
4:   for each gradient step do
5:       Compute the maximum absolute TD error $\|\epsilon_{TD}\|_{\infty}$ in a minibatch.
6:       Update $\lambda$ and $\lambda'$ using Eq. (13).
7:       Update $\theta$ with one step of SGD using Eq. (11)
8:   end for
9:  end for

5. Experiments

This section empirically studies the proposed GVI with tabular and deep implementation. We wanted to evaluate the effectiveness of our error-aware KL coefficient design in handling the trade-off between learning speed and stability. For didactic purposes, we first evaluated GVI on a tabular maze environment that is the same as the one used in Figure 1. The tabular experiments serve to verify that GVI can better handle the trade-off problem between learning speed and robustness than the constant KL coefficient scheme. We then conducted an experiment on classic control tasks from OpenAI Gym benchmarks (Brockman et al., 2016) to observe the behavior of GVI with deep implementation. For the deep RL experimentation, we consider GVI as a variation of Munchausen-DQN (M-DQN) (Vieillard et al., 2020c) and thus take M-DQN as our baseline.

Tabular Experiments Figure 3 investigates the optimality gap of GVI with varying conditions. For GVI, we also included the investigation of the introduced hyperparameters $\alpha_1$ and $\alpha_2$ and their impact on performance. Although they do not play any role in error analysis, in practice they can have a large effect on the trade-off between speed and stability.

The left graph in Figure 3 compares the best behavior of GVI with MD-VI, where the parameters of GVI are fine-tuned to yield the empirically best performance. The figure shows that GVI achieves faster and more robust convergence than MD-VI under a certain hyperparameter. GVI reaches the minimum optimality gap in around 50 steps and keeps the value under $10^{-3}$. On the other hand, MD-VI suffers from the trade-off between speed and stability. While $\lambda = 50$ reaches the minimum optimality gap close to that of GVI, it reaches it in around 1000 steps and is thus much slower than GVI. MD-VI with $\lambda = 30$ converges faster, but the optimality gap oscillates and exceeds $10^{-2}$. Therefore, it can be safely concluded that the constant KL coefficient scheme MD-VI cannot outperform GVI.

The middle and the right graphs are plotted to investigate the behavior of GVI with different $\alpha_1$ and $\alpha_2$. GVI with a small $\alpha_1$ never reaches the optimal value, while experiments with the small $\alpha_2$ obtain a small optimality gap at the cost of huge oscillation. These are expected since $\lambda_k$ corresponds to the learning rate of the updates (Kozuno et al., 2019), and $\alpha_2$ decides how long the conservativeness remains after detecting large errors.

Deep RL Experiments Using a set of classic control benchmarks (Brockman et al., 2016), we examine the DGVI of Algorithm 1 against the constant KL coefficient algorithm of M-DQN (Vieillard et al., 2020c). We choose the LunarLander-v2, CartPole-v1,
and Pendulum-v0 environments as our benchmarks. Since our DGVI and M-DQN support only discrete action space environments, we discretized the continuous action space of Pendulum-v0 into five discrete actions. For each seed we perform 10 evaluation rollouts every 300 environment steps. For a fair comparison, all of the algorithms share the same hyperparameters except the KL regularization. To highlight the robustness of algorithms against estimation errors, we explicitly remove target networks from the algorithms, even though such networks provide a key ingredient to the success of modern deep RL (Mnih et al., 2015; Haarnoja et al., 2018). All figures are plotted by averaging results from five independent random seeds for statistical results. We list the set of hyperparameters in Appendix C.

Figure 4 shows the learning curves of algorithms and the corresponding KL regularization of DGVI. Compared to the constant regularized algorithms, GVI achieves more stable learning in DiscretePendulum and CartPole. Notably, GVI has smaller regularization in DiscretePendulum and CartPole than $\lambda = 10$. This indicates that the dynamic change of the KL coefficient is more important than its magnitude.

To observe how the dynamic KL coefficient improves stability, we evaluated the maximum absolute TD error $\|\epsilon_{TD}\|_{\infty}$ as shown in Figure 5. Compared to constant regularized algorithms, the error of DGVI proves to be much smaller during learning. This result agrees well with how DGVI updates the network by Eq. (11): the bootstrap is scaled by $\lambda'$, which becomes small when DGVI encounters large errors. For a better understanding of the effect on the bootstrap, consider an extreme case where a significantly huge error is induced and $\lambda'$ is infinite. Then, the loss becomes $L_{\theta} \approx \mathbb{E}_{(s,a) \sim B} [(q_{\theta}(s,a) - \ln \pi(s,a))^2] = \mathbb{E}_{s \sim B} \left[ (\ln \sum_{a \in A} \exp(q_{\theta}(s,a)))^2 \right]$, and thus the new $q_{\theta}$ will have smaller values. GVI thus tends to underestimate the state and action pairs where huge errors are expected, which is assumed to prevent bad updates from quickly spreading to downstream $Q$-values. We can conclude that the proposed mechanism renders DGVI stable even without target networks.

6. Related Work and Discussion

The recent boom in the literature on KL-regularized ADP (Azar et al., 2012; Ghavamzadeh et al., 2011; Bellemare et al., 2016; Vieillard et al., 2020d; Kozuno et al., 2019) has demonstrated the effectiveness of KL regularization against estimation errors. The most relevant algo-
rithms to our proposed approach are Mirror Descent Value Iteration (MD-VI) (Geist et al., 2019) and Munchausen Value Iteration (MVI) (Vieillard et al., 2020c); those algorithms introduce a KL penalty on both the greedy and the evaluation steps. While some studies have focused on the error bounds of KL-regularized AVI (Vieillard et al., 2020a), how dynamic changes in the regularization coefficient affect performance has been left largely untouched. To the best of our knowledge, this paper is the first work to provide the error bound of an AVI with dynamic KL regularization.

While it has not been discussed in the ADP literature, dynamic KL regularization has appeared in many deep RL algorithms. Dynamic KL regularization is often introduced to restrict aggressive policy improvement steps. Trust region policy optimization
(TRPO) (Schulman et al., 2015) is one such seminal algorithm that introduces KL constraints to approximately ensure monotonic improvement. Based on TRPO, many algorithms leverage the KL constraint and demonstrate promising performance on challenging environments (Schulman et al., 2017; Nachum et al., 2018; Abdolmaleki et al., 2018), and Nachum et al. (2018) introduced a dynamic KL coefficient design to create a trust region. However, the above-mentioned algorithms design the dynamic coefficient based on heuristics, while we design it by leveraging rigorous analysis as shown in Theorem 2. Furthermore, trust-region methods consider the KL constraints even when there are no estimation errors, and thus they may overly slow down learning.

In addition to the dynamic KL regularization, DGVI has an important feature: error awareness. One of the most well-known algorithms making use of TD error is Prioritized Experience Replay (PER) (Schaul et al., 2015). PER utilizes TD error for prioritizing the samples in the replay buffer to increase the appearance of rare samples. On the other hand, DGVI mitigates the effect of rare samples that may have huge TD errors by scaling its bootstrapping. Thus, slower learning will be expected when exploration matters: the rare samples will have less of an affect than usual in DGVI. We do not consider this problem as exploration that is out of our scope.

In this work, we do not consider Shannon entropy for regularization. Some entropy regularized ADP literature has established that by augmenting the reward with Shannon entropy, the optimal policy becomes multi-modal and hence robust against adversarial settings (Haarnoja et al., 2017, 2018; Ahmed et al., 2019). We leave GVI with Shannon entropy regularization as future work due to the complex theoretical analysis.

7. Conclusion

We have presented the first error-aware KL coefficient design for RL algorithms and developed a novel error-aware RL algorithm, Geometric Value Iteration (GVI), which features a dynamic error-aware KL coefficient design aimed at mitigating the impact of errors on performance. The theoretical error bound analysis provides two guidelines for efficient learning: the coefficient should be increased when a large error is induced but its effect should not be overly large. This dynamic regularization allows GVI to address the trade-off problem between robustness and convergence speed, which has been largely left untouched in previous ADP studies.

In addition to GVI as an ADP scheme, we further combined GVI with deep networks. Based on the recent framework introduced by Vieillard et al. (2020c), we implement GVI as a deep RL algorithm, and the resulting algorithm, deep GVI (DGVI), achieves robustness against errors by reducing the bootstrapping effect when it meets huge TD errors. Our experiments verified not only the faster and more stable learning of GVI but also the more robust learning of DGVI even without target networks.

While our algorithm can be easily applied to standard deep RL frameworks, our empirical studies are limited to classic control tasks due to the expensive computational cost of recent Deep RL benchmarks, e.g., Atari games (Bellemare et al., 2013). We believe that the classic control tasks are sufficient to verify our algorithms and thus leave evaluation on a set of high-dimensional benchmarks as future work.
Acknowledgments

This work is partly supported by JSPS KAKENHI Grant Number 21H03522 and 21J15633.

References


