REGQ: CONVERGENT Q-LEARNING WITH LINEAR FUNCTION APPROXIMATION USING REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Q-learning is widely used algorithm in reinforcement learning community. Under the lookup table setting, its convergence is well established. However, its behavior is known to be unstable with the linear function approximation case. This paper develops a new Q-learning algorithm, called RegQ, that converges when linear function approximation is used. We prove that simply adding an appropriate regularization term ensures convergence of the algorithm. We prove its stability using a recent analysis tool based on switching system models. Moreover, we experimentally show that RegQ converges in environments where Q-learning with linear function approximation has known to diverge. We also provide an error bound on the solution where the algorithm converges.

1 INTRODUCTION

Recently, reinforcement learning has shown great success in various fields. For instance, Mnih et al., 2015 achieved human level performance in several video games in the Atari benchmark (Bellemare et al., 2013). Since then, researches on deep reinforcement learning algorithms have shown significant progresses (Lan et al., 2020; Chen et al., 2021). For example, Badia et al., 2020 performs better than standard human performance in all 57 Atari games. Schrittwieser et al., 2020 solves Go, chess, Shogi, and Atari without prior knowledge about the rules. Although great success has been achieved in practice, there is still gap between theory and the practical success. Especially when off-policy, function approximation, and bootstrapping are used together, the algorithm may diverge or show unstable behaviors. This phenomenon is called the deadly triad (Sutton & Barto, 2018). Famous counter-examples are given in Baird, 1995; Tsitsiklis & Van Roy, 1997.

For policy evaluation, especially for temporal-difference (TD) learning algorithm, there has been several algorithms to resolve the deadly triad issue. Bradtke & Barto, 1996 uses the least-square method to compute a solution of TD-learning, but it suffers from $O(h^2)$ time complexity, where h is number of features. Maei, 2011; Sutton et al., 2009 developed gradient descent based methods which minimize the mean square projected Bellman error. Ghiassian et al., 2020 added regularization term to TD Correction (TDC) algorithm, which uses single time scale step-size. Lee et al., 2021 introduced several variants of the gradient TD (GTD) algorithm under control theoretic frameworks. Sutton et al., 2016 re-weights some states to match on-policy distribution to stabilize the off-policy TD-learning. Diddigi et al., 2019 uses l_2 regularization to propose a new convergent off-policy TD-learning algorithm. Mahadevan et al., 2014 studied regularization on off-policy TD-learning through the lens of primal dual method.

First presented by Watkins & Dayan, 1992, Q-learning also suffers from divergence issues under the deadly triad. While there are convergence results under the look-up table setting (Watkins & Dayan, 1992; Jaakkola et al., 1994; Borkar & Meyn, 2000; Lee & He, 2019), even with simple linear function approximation, the convergence is only guaranteed under strong assumptions (Melo et al., 2008; Lee & He, 2019; Yang & Wang, 2019). Melo et al., 2008 adopts an assumption on relation between behavior policy and target policy to guarantee convergence, which is not practical in general. Lee & He, 2019 assumes a strong assumption to ensure the convergence with the so-called switching system approach. Yang & Wang, 2019 has a stringent assumption on anchor state-action pairs. Devraj & Meyn, 2017 proposes a Q-learning algorithm that minimizes asymptotic variance. However, it requires the assumption that number of changes of policy are finite, and involves matrix inversion at each iteration. There are few works, (Agarwal et al., 2021; Carvalho et al., 2020; Zhang et al., 2021; Maei et al., 2010), that guarantee convergence under more general assumptions.

The main goal of this paper is to propose a practical Q-learning algorithm, called regularized Q-learning (RegQ), that guarantees convergence under linear function approximation. We prove its convergence using the ordinary differential equation (O.D.E) analysis framework in (Borkar & Meyn, 2000) together with the switching system approach developed in Lee & He, 2019. As in Lee & He, 2019, we construct upper and lower comparison systems, and prove its global asymptotic stability based on switching system theories. Compared to the standard Q-learning in (Watkins & Dayan, 1992), the only difference is the additional l_2 regularization term, which makes the algorithm relevantly simple. Compared to the previous works in Carvalho et al. (2020); Maei et al. (2010), our algorithm is single time-scale, and hence, shows faster convergence rates experimentally. Our algorithm directly uses bootstrapping rather than circumventing the issue in the deadly triad. Therefore, it could give a new insight into training reinforcement learning algorithms without using the so-called target network technique introduced in Mnih et al., 2015. The main contributions of this paper are summarized as follows:

1. A new single time-scale Q-learning algorithm with linear function approximation is proposed.
2. We prove the convergence of the proposed algorithm based on the O.D.E approach together with the switching system model in Lee & He, 2019.
3. We experimentally show that our algorithm performs faster than other two time-scale Q-learning algorithms in Carvalho et al. (2020); Maei et al. (2010).

Related works are summarized as follows: Motivated by the empirical success of the deep Q-learning in Mnih et al., 2015, recent works in Zhang et al., 2021; Carvalho et al., 2020; Agarwal et al., 2021 use the target network to circumvent the bootstrapping issue and guarantees convergence. Carvalho et al., 2020 uses a two time-scale learning method, and has a strong assumption on the boundedness of the feature matrix. Zhang et al., 2021 uses l_2 regularization with the target network, while a projection step is involved, which makes it difficult to implement practically. Moreover, it also uses a two time-scale learning method. Agarwal et al., 2021 additionally uses the so-called experience replay technique with the target network, and also has a strong assumption on the boundedness of the feature matrix. Furthermore, the optimality is only guaranteed under a specific type of Markov Decision Process. Maei et al., 2010 suggested the so-called Greedy-GQ (gradient Q-learning) algorithm, but due to non-convexity of the objective function, it could converge to a local optima.

2 Preliminaries and Notations

2.1 Markov Decision Process

We consider an infinite horizon Markov Decision Process (MDP), which consists of a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where the state space $\mathcal{S}$ and action space $\mathcal{A}$ are finite sets, P denotes the transition probability, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward, and $\gamma \in (0,1)$ is the discount factor. Given a stochastic policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, where $\mathcal{P}(\mathcal{A})$ is the set of probability distributions over $\mathcal{A}$, agent at the current state s_k selects an action $a_k \sim \pi(\cdot|s_k)$, then the agent's state changes to the next state $s_{k+1} \sim P(\cdot|s_k, a_k)$, and receives reward $r_{k+1} := r(s_k, a_k, s_{k+1})$. A deterministic policy is a special stochastic policy, which can be defined simply as a mapping $\pi : \mathcal{S} \to \mathcal{A}$, which maps a state to an action.

The objective of MDP is to find a deterministic optimal policy, denoted by π^*, such that the cumulative discounted rewards over infinite time horizons is maximized, i.e., $\pi^* := \arg\max_\pi \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_k \middle| \pi\right]$, where $(s_0, a_0, s_1, a_1, \ldots)$ is a state-action trajectory generated by the Markov chain under policy π, and $\mathbb{E}[\cdot|\pi]$ is an expectation conditioned on the policy π. The Q-function under policy π is defined as $Q^\pi(s,a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_k \middle| s_0 = s, a_0 = a, \pi\right], \; s \in \mathcal{S}, \; a \in \mathcal{A}$, and the optimal Q-function is defined as $Q^*(s,a) = Q^{\pi^*}(s,a)$ for all $s \in \mathcal{S}, a \in \mathcal{A}$. Once Q^* is known, then an optimal policy can be retrieved by the greedy action, i.e., $\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s,a)$. Throughout, we assume that the MDP is ergodic so that the stationary state distribution exists and the MDP is well posed, which is standard in the literature.

It is known that the optimal Q-function satisfies the so-called Bellman equation expressed as follows:

$$Q^*(s,a) = \mathbb{E}\left[r_{k+1} + \max_{a_{k+1}\in\mathcal{A}} \gamma Q^*(s_{k+1},a_{k+1}) \middle| s_k = s, a_k = a \right] := \mathcal{T}Q^* \tag{1}$$

where $\mathcal{T}$ is called the Bellman operator.

2.2 Notations

In this paper, we will use an O.D.E. model (Borkar & Meyn, 2000) of Q-learning to analyze its convergence. To this end, it is useful to introduce some notations in order to simplify the overall expressions. Throughout the paper, e_a and e_s denote a-th and s-th canonical basis vectors in $\mathbb{R}^{|\mathcal{A}|}$ and $\mathbb{R}^{|\mathcal{S}|}$, respectively. Moreover, $\otimes$ stands for the Kronecker product. Let us introduce the following notations:

$$P := \begin{bmatrix} P_1 \\ \vdots \\ P_{|\mathcal{A}|} \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}|}, \quad R := \begin{bmatrix} R_1 \\ \vdots \\ R_{|\mathcal{A}|} \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}, \quad Q := \begin{bmatrix} Q_1 \\ \vdots \\ Q_{|\mathcal{A}|} \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|},$$

$$D_a := \begin{bmatrix} d(1,a) & & \\ & \ddots & \\ & & d(|\mathcal{S}|,a) \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}, \quad D := \begin{bmatrix} D_1 & & \\ & \ddots & \\ & & D_{|\mathcal{A}|} \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|},$$

where $P_a \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}, a \in \mathcal{A}$ is the state transition matrix whose i-th row and j-th column component denotes the probability of transition to state j when action a is taken at state i, $P^\pi \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|}$ represents the state-action transition matrix under policy π, i.e.,

$$(e_s \otimes e_a)^T P^\pi (e_{s'} \otimes e_{a'}) = \mathbb{P}[s_{k+1} = s', a_{k+1} = a' | s_k = s, a_k = a, \pi],$$

$Q_a = Q(\cdot, a) \in \mathbb{R}^{|\mathcal{S}|}, a \in \mathcal{A}$ and $R_a(s) := \mathbb{E}[r(s,a,s')|s,a], s \in \mathcal{S}$. Moreover, $d(\cdot,\cdot)$ is the state-action visit distribution, where i.i.d random variables $\{(s_k,a_k)\}_{k=0}^{\infty}$ are sampled, i.e., $d(s,a) = \mathbb{P}[s_k = s, a_k = a],\ a \in \mathcal{A}, s \in \mathcal{S}$. With a slight abuse of notation, d will be also used to denote the vector $d \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ such that $d^T(e_s \otimes e_a) = d(s,a),\ \forall s \in \mathcal{S}, a \in \mathcal{A}$. In this paper, we represent a policy in a matrix form in order to formulate a switching system model. In particular, for a given policy π, define the matrix

$$\Pi^\pi := \begin{bmatrix} (e_{\pi(1)} \otimes e_1)^T & \cdots & (e_{\pi(|\mathcal{S}|)} \otimes e_{|\mathcal{S}|})^T \end{bmatrix}^T \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}||\mathcal{A}|}. \tag{2}$$

Then, we can prove that for any deterministic policy, π, we have $\Pi^\pi Q = \begin{bmatrix} Q(1,\pi(1))^T & Q(2,\pi(2))^T & \cdots & Q(|\mathcal{S}|,\pi(|\mathcal{S}|))^T \end{bmatrix}^T$. For simplicity, let $\Pi_Q := \Pi^\pi$ when $\pi(s) = \arg\max_{a\in\mathcal{A}} Q(s,a)$. Moreover, we can prove that for any deterministic policy π, $P^\pi = P\Pi^\pi \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|}$, where P^π is the state-action transition probability matrix. Using the notations introduced, the Bellman equation in (1) can be compactly written as $Q^* = \gamma P \Pi_{Q^*} Q^* + R =: \mathcal{T}Q^*$, where π_{Q^*} is the greedy policy defined as $\pi_{Q^*}(s) = \arg\max_{a\in\mathcal{A}} Q^*(s,a)$.

2.3 Q-learning with linear function approximation

Q-learning is widely used model-free learning to find Q^*, whose updates are given as

$$Q_{k+1}(s_k,a_k) \leftarrow Q_k(s_k,a_k) + \alpha_k \delta_k, \tag{3}$$

where $\delta_k = r_{k+1} + \gamma \max_{a\in\mathcal{A}} Q_k(s_{k+1},a) - Q_k(s_k,a_k)$ is called the TD error. Each update uses an i.i.d. sample $(s_k, a_k, r_{k+1}, s_{k+1})$, where (s_k,a_k) is sampled from a state-action distribution $d(\cdot,\cdot)$.

Here, we assume that the step-size is chosen to satisfy the so-called the Robbins-Monro condition (Robbins & Monro, 1951), $\alpha_k > 0$, $\sum\limits_{k=0}^{\infty} \alpha_k = \infty$, $\sum\limits_{k=0}^{\infty} \alpha_k^2 < \infty$. When the state-spaces and action-spaces are too large, then the memory and computational complexities usually become intractable. In such cases, function approximation is commonly used to approximate Q-function (Mnih et al., 2015; Schrittwieser et al., 2020; Hessel et al., 2018; Lan et al., 2020). Linear function approximation is one of the simplest function approximation approaches. In particular, we use the feature matrix $X \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times h}$ and parameter vector $\theta \in \mathbb{R}^h$ to approximate Q-function, i.e., $Q \simeq X\theta$, where the feature matrix is expressed as $X := \begin{bmatrix} x(1,1)^T & \cdots & x(1,|\mathcal{A}|)^T & \cdots & x(|\mathcal{S}|,|\mathcal{A}|)^T \end{bmatrix}^T \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times h}$. Here, $x(\cdot,\cdot) \in \mathbb{R}^h$ is called the feature vector, and h is a positive integer with $h \ll |\mathcal{S}||\mathcal{A}|$. The corresponding greedy policy becomes $\pi_{X\theta}(s) = \arg\max_{a\in\mathcal{A}} x(s,a)^T\theta$. Note that the number of policies characterized by the greedy policy is finite. This is because the policy is invariant under constant multiplications, and there are a finite number of sectors on which the policy is invariant. This is also common in the literature (Maei et al., 2010; Zhang et al., 2021; Carvalho et al., 2020). Next, we summarize some standard assumptions adapted throughout this paper.

Assumption 2.1. *The state-action visit distribution is positive, i.e., $d(s,a) > 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$.*

Assumption 2.2. *The feature matrix, X, has full column rank, and is a non-negative matrix. Moreover, columns of X are orthogonal.*

Assumption 2.3 (Boundedness on feature matrix and reward matrix)**.** *There exists constants, $X_{\max} > 0$ and $R_{\max} > 0$, such that $\max(||X||_\infty, ||X^T||_\infty) < X_{\max}, \quad ||R||_\infty < R_{\max}$.*

Note that Assumption 2.1, Assumption 2.2 and Assumption 2.3 are commonly adopted in the literature, e.g. Carvalho et al. (2020); Melo et al. (2008); Lee & He (2019). Moreover, under Assumption 2.1, D is a nonsingular diagonal matrix with strictly positive diagonal elements.

Lemma 2.1 (Gosavi (2006))**.** *Under Assumption 2.3, the optimal Q-function, Q^*, is bounded, i.e., $||Q^*||_\infty \leq \frac{R_{\max}}{1-\gamma}$.*

The proof of Lemma 2.1 comes from the fact that under the discounted infinite horizon setting, Q^* can be expressed as an infinite sum of a geometric sequence.

Remark 2.1. *Carvalho et al., 2020; Agarwal et al., 2021 assume $||x(s,a)||_\infty \leq 1$ for all $(s,a) \in \mathcal{S}\times\mathcal{A}$. Moreover, Zhang et al. 2021 requires specific bounds on the feature matrix which is dependent on various factors e.g. projection radius and transition matrix . On the other hand, our feature matrix can be chosen arbitrarily large regardless of those factors.*

2.4 O.D.E. Analysis

The dynamic system framework has been widely used to prove convergence of reinforcement learning algorithms, e.g., Sutton et al. 2009; Maei et al. 2010; Borkar & Meyn 2000; Lee & He 2019; Carvalho et al. 2020; Lee et al. 2021. Especially, Borkar & Meyn, 2000 is one of the most widely used techniques to prove stability of stochastic approximation using O.D.E. analysis. Consider the following stochastic algorithm with a non-linear mapping $f : \mathbb{R}^n \to \mathbb{R}^n$:

$$\theta_{k+1} = f(\theta_k) + m_k, \tag{4}$$

where $m_k \in \mathbb{R}^n$ is an i.i.d. noise vector. For completeness, results in Borkar & Meyn, 2000 are briefly reviewed in the sequel. Under the Assumption A.1 given in the Appendix, we now introduce Borkar and Meyn theorem below.

Lemma 2.2 (Borkar and Meyn theorem)**.** *Suppose that Assumption A.1 in the Appendix holds, and consider the stochastic algorithm in (4). Then, for any initial $\theta_0 \in \mathbb{R}^n$, $\sup_{k\geq 0} ||\theta_k|| < \infty$ with probability one. In addition , $\theta_k \to \theta^e$ as $k \to \infty$ with probability one, where $f(\theta^e) = 0$.*

The main idea of Borkar and Meyn theorem is as follows: iterations of a stochastic recursive algorithm follow the solution of its corresponding O.D.E. in the limit when the step-size satisfies the Robbins-Monro condition. Therefore, by proving the asymptotic stability of the O.D.E., we can induce the convergence of the original algorithm. In this paper, we will use an O.D.E. model of Q-learning, which is expressed as a special nonlinear system called a switching system. In the next section, basic concepts in switching system theory are briefly introduced.

2.5 Switching System

In this paper, we will consider a particular nonlinear system, called the *switched linear system* (Liberzon, 2003),

$$\dot{x}_t = A_{\sigma_t} x_t, \quad x_0 = z \in \mathbb{R}^n, \quad t \in \mathbb{R}_+, \tag{5}$$

where $x_t \in \mathbb{R}^n$ is the state, $\mathcal{M} := \{1, 2, \ldots, M\}$ is called the set of modes, $\sigma_t \in \mathcal{M}$ is called the switching signal, and $\{A_\sigma, \sigma \in \mathcal{M}\}$ are called the subsystem matrices. The switching signal can be either arbitrary or controlled by the user under a certain switching policy. Especially, a state-feedback switching policy is denoted by $\sigma(x_t)$.

Stability and stabilization of (5) have been widely studied for decades. Still, finding a practical and effective condition for them is known to be a challenging open problem. For example, contrary to linear time-invariant systems, even if each subsystem matrix A_σ is Hurwitz, the overall switching system may not be stable in general. This tells us that tools in linear system theories cannot be directly applied to conclude the stability of the switching system.

Another approach is to use the Lyapunov theory (Khalil, 2002). From standard results in control system theories, finding a Lyapunov function ensures stability of the switching system. If the switching system consists of negative definite matrices, then we can always find a common quadratic Lyapunov function and ensure its stability. We will use this fact to prove the convergence of the proposed algorithm. In particular, the proposed Q-learning algorithm can be modelled as a switching system, whose subsystem matrices are all negative definite.

3 Projected Bellman equation

In this section, we introduce the notion of projected Bellman equation with a regularization term, and establish connections between it and the proposed algorithm. Moreover, we briefly discuss the existence and uniqueness of the solution of the projected Bellman equation. We will also provide an example to illustrate the existence and uniqueness. When using the linear function approximation, since the true action value may not lie in the subspace spanned by the feature vectors, a solution of the Bellman equation may not exist in general. To resolve this issue, a standard approach is to consider the projected Bellman equation defined as

$$X\theta^* = \Gamma\mathcal{T}X\theta^*, \tag{6}$$

where $\Gamma := X(X^TDX)^{-1}X^TD$ is the weighted Euclidean Projection with respect to state-action visit distribution onto the subspace spanned by the feature vectors, and $\mathcal{T}X\theta^* = \gamma P\Pi_{X\theta^*}X\theta^* + R$. In this case, there is more chances for a solution satisfying the above projected Bellman equation to exist. Still, there may exist cases where the projected Bellman equation does not admit a solution. We will give an example of such case in Appendix A.9. To proceed, let us rewrite (6) equivalently as

$$X\theta^* = X(X^TDX)^{-1}X^TD(\gamma P\Pi_{X\theta^*}X\theta^* + R) \Leftrightarrow \underbrace{(X^TDX - \gamma X^TDP\Pi_{X\theta^*}X)}_{A_{\pi_{X\theta^*}}}\theta^* = \underbrace{X^TDR}_{b},$$

where we use the simplified notations $A_{\pi_{X\theta^*}} := X^TDX - \gamma X^TDP\Pi_{X\theta^*}X,\ b = X^TDR$. Therefore, the projected Bellman equation in (6) can be equivalently written as the nonlinear equation

$$b - A_{\pi_{X\theta^*}}\theta^* = 0. \tag{7}$$

A potential deterministic algorithm to solve the above equation is

$$\theta_{k+1} = \theta_k + \alpha_k(b - A_{\pi_{X\theta_k}}\theta_k). \tag{8}$$

If it converges, i.e., $\theta_k \to \theta^*$ as $k \to \infty$, then it is clear that θ^* solves (7). In this paper, the proposed algorithm is a stochastic algorithm that solves the modified equation

$$b - (A_{\pi_{X\theta_e}} + \eta C)\theta_e = 0, \tag{9}$$

where $C := X^TDX$, and $\eta \geq 0$ is a weight on the regularization term. Similar to (8), the corresponding deterministic algorithm is

$$\theta_{k+1} = \theta_k + \alpha_k(b - (A_{\pi_{X\theta_k}} + \eta C)\theta_k). \tag{10}$$

If it converges, i.e., $\theta_k \to \theta_e$ as $k \to \infty$, then it is clear that θ_e solves (9). Some natural questions that arise here are as follows: Which conditions can determine the existence and uniqueness of the equations in (7) and (9)? Partial answers are given in the sequel. Considering the non-existence of fixed point of (6) (De Farias & Van Roy, 2000), both (7) and (9) may not also have a solution. However, for the modified Bellman equation in (9), we can prove that under appropriate conditions, its solution exists and is unique. We give an example where the solution does not exist for (7) but does exist for (9) in Appendix A.9.

Lemma 3.1. *When $\eta > ||(X^T DX)^{-1}||_\infty X_{\max}^2 - 1$, a solution of (9) exists and is unique.*

The proof is given in Appendix A.3, and uses Banach fixed-point theorem (Agarwal et al., 2018). The proof follows similar lines as in Melo (2001) to establish the existence and uniqueness of solution of (1). From Lemma 3.1, we can see that when the weight η is sufficiently large, the existence and uniqueness of the solution is guaranteed. Note that even if a solution satisfying (9) exists, $X\theta_e$ may be different from the optimal Q-function, Q^*. However, we can derive a bound on the error, $X\theta_e - Q^*$, using some algebraic inequalities and contraction property of Bellman operator, which is presented below.

Lemma 3.2. *Assume that a solution of (9) exists. Denoting $\Gamma := X(X^T DX)^{-1}DX$ as the projection operator, when $\eta > \gamma||\Gamma||_\infty - 1$, we have the following bound:*

$$||X\theta_e - Q^*||_\infty \leq \frac{1}{1+\eta-\gamma||\Gamma||_\infty}||Q^* - \Gamma Q^*||_\infty + \frac{\eta}{1+\eta-\gamma||\Gamma||_\infty}\frac{R_{\max}}{1-\gamma}.$$

Some remarks are in order for Lemma 3.2. First of all, $\eta > \gamma||\Gamma||_\infty - 1$ ensures that the error is always bounded. The first term represents the error incurred by the difference between the optimal Q^* and Q^* projected onto the feature space. Therefore, this error is induced by the linear function approximation. The second term represents the error potentially induced by the regularization. For instance, if $1 > \gamma||\Gamma||_\infty$, then the second error term vanishes as $\eta \to 0$. Finally, note that as $\eta \to \infty$, the first error term vanishes. This means that the error by the regularization dominates the error induced by the linear function approximation.

4 Algorithm

In this section, we will introduce our main algorithm, called RegQ, and elaborate the condition on the regularization term to make the algorithm convergent. The proposed algorithm is motivated by TD-learning. In particular, for on-policy TD-learning, one can establish its convergence using the property of the stationary distribution. On the other hand, for an off-policy case, the mismatch between the sampling distribution and the stationary distribution could cause its divergence (Sutton et al., 2016). To address this problem, Diddigi et al., 2019 adds a regularization term to TD-learning in order to make it convergent. Since Q-learning can be interpreted as an off-policy TD-learning, we add a regularization term to Q-learning update motivated by Diddigi et al., 2019. This modification leads to the proposed RegQ algorithm as follows:

$$\theta_{k+1} = \theta_k + \alpha_k(x(s_k, a_k)\delta_k + \eta x(s_k, a_k)x(s_k, a_k)^T\theta_k) \tag{11}$$

Note that letting $\eta = 0$, the above update is reduced to the standard Q-learning with linear function approximation in (3). The proposed RegQ is different from Diddigi et al., 2019 in the sense that a regularization term is applied to Q-learning instead of TD-learning. Rewriting the stochastic update in a deterministic manner, it can be written as follows:

$$\theta_{k+1} = \theta_k + \alpha_k(b - (A_{\pi_{X\theta_k}} + \eta C)\theta_k + m_{k+1}), \tag{12}$$

where $m_{k+1} = \delta_k x(s_k, a_k) + \eta x(s_k, a_k)^T x(s_k, a_k)\theta_k - (b - (A_{\pi_{X\theta_k}} + \eta C)\theta_k)$ is an i.i.d. noise. Note that without the noise, (12) is reduced to the deterministic version in (10). In our convergence analysis, we will apply the O.D.E. approach, and in this case, $A_{\pi_{X\theta_k}} + \eta C$ will determine the stability of the corresponding O.D.E. model, and hence, convergence of (11). Note that (12) can be interpreted as a switching system defined in (5) with stochastic noises. As mentioned earlier, proving the stability of a general switching system is challenging in general. However, we can find a common Lyapunov function to prove its asymptotic stability. In particular, we can make $-(A_{\pi_{X\theta_k}} + \eta C)$ to be negative definite under the following condition:

$$\eta > \max_{\pi\in\Theta, s\in\mathcal{S}, a\in\mathcal{A}} \frac{\gamma d^T P^\pi(e_a \otimes e_s)}{2d(s,a)} - \frac{2-\gamma}{2}, \tag{13}$$

where Θ is the set of all deterministic policies, and $\otimes$ is the Kronecker product. Lemma A.2, given in Appendix A.2, is similar to Theorem 2 in Diddigi et al., 2019, and ensures such a property. Now, we can use the Lyapunov argument to establish stability of the overall system. Building on the negative definiteness of the $-(A_{\pi_{X\theta_k}} + \eta C)$, in the next section, we prove that under the stochastic update (11), we have $\theta_k \to \theta_e$ as $k \to \infty$ with probability one, where θ_e satisfies the projected Bellman equation in (9). If $\eta = 0$ satisfies (13), we can guarantee convergence to an optimal policy without errors.

5 Convergence Analysis

Recently, Lee & He, 2019 suggested a switching system framework to prove the stability of Q-learning in the linear function approximation cases. However, its assumption seems too stringent to check in practice. Here, we develop more practical Q-learning algorithm by adding an appropriately preconditioned regularization term. We prove the convergence of the proposed Q-learning with regularization term (11) following lines similar to Lee & He, 2019. Our proof mainly relies on Borkar-Meyn theorem. Therefore, we first discuss about the corresponding O.D.E. for the proposed update in (11), which is

$$\dot{\theta}_t = -(1+\eta)X^T DX\theta_t + \gamma X^T DP\Pi_{X\theta_t}X\theta_t + X^T DR := f(\theta_t). \tag{14}$$

Then, using changes of coordinates, the above O.D.E. can be rewritten as

$$\frac{d}{dt}(\theta_t - \theta_e) = (-(1+\eta)X^T DX + \gamma X^T DP\Pi_{X\theta_t}X)(\theta_t - \theta_e) + \gamma X^T DP(\Pi_{X\theta_t} - \Pi_{X\theta_e})X\theta_e, \tag{15}$$

where θ_e satisfies (9). Here, we assume that the equilibrium point exists and is unique. We later prove that if the equilibrium exists, then it is unique. To apply Borkar-Meyn theorem, we discuss about the asymptotic stability of the O.D.E. in (15), and check conditions of Assumption A.1 in Appendix A.5. Note that (15) includes an affine term, i.e., it cannot be expressed as a matrix times $\theta_t - \theta_e$. Establishing asymptotic stability of switched linear system with affine term is difficult compared to switched linear system (5). To circumvent this difficulty, Lee & He, 2019 proposed upper and lower systems, which upper bounds and lower bounds the original system, respectively using the so-called vector comparison principle. Then, the stability of the original system can be established by proving the stability of the upper and lower systems, which are easier to analyze. Following similar lines, to check global asymptotic stability of the original system, we also introduce upper and lower systems, which upper bounds and lower bounds the original system, respectively. Then, we prove global asymptotic stability of the two bounding systems. Since upper and lower systems can be viewed as switched linear system and linear system, respectively, the global asymptotic stability is easier to prove. We stress that although the switching system approach in Lee & He, 2019 is applied in this paper, the detailed proof is entirely different and nontrivial. In particular, the upper and lower systems are given as follows:

$$\frac{d}{dt}\theta_t^u = (-(1+\eta)X^T DX + \gamma X^T DP\Pi_{X\theta_t^u}X)\theta_t^u$$
$$\frac{d}{dt}\theta_t^l = -((1+\eta)X^T DX + \gamma X^T DP\Pi_{X\theta_e}X)\theta_t^l,$$

where θ_t^u denotes the state of the upper system, and θ_t^l stands for the state of the lower system. We defer the detailed construction of each system to Appendix A.6. Establishing stability of upper and lower system gives the stability of overall system.

Theorem 5.1. *Suppose that (a) Assumption 2.2 holds, (b) (13)) holds, and (c) a solution of (9) exists. Then, the solution is unique, and the origin is the unique globally asymptotically stable equilibrium point of (15).*

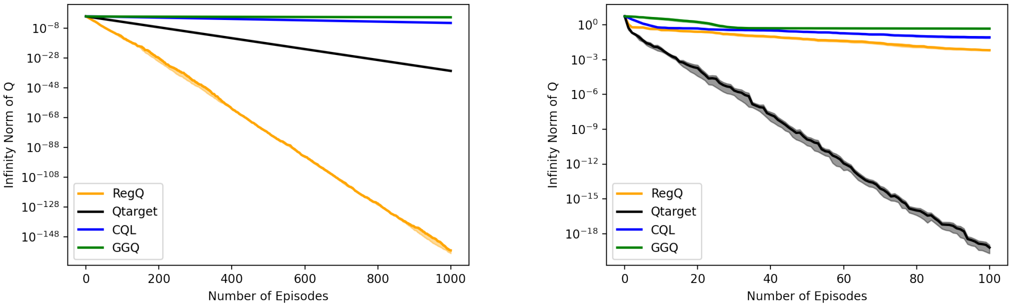
The detailed proof is given in Appendix A.6.

Building on previous results, we now use Borkar and Meyn's theorem Lemma 2.2 to establish the convergence of RegQ. The full proof of the following theorem is given in Appendix A.7.

Theorem 5.2. *If η satisfies (13). Then under Assumption 2.1, Assumption 2.2 and Assumption 2.3, under the stochastic update (11), $\theta_k \to \theta_e$ as $k \to \infty$ with probability one, where θ_e satisfies (9).*

6 Experiments

In this section, we present experimental results under well-known environments in Tsitsiklis & Van Roy (1996); Baird (1995), where Q-learning with linear function approximation diverges. We also compare performance under the Mountain Car environment (Sutton & Barto, 2018) where Q-learning performs well. In Appendix A.8.2, we show experimental results under various step-size and η. We also show trajectories of upper and lower systems to illustrate the theoretical results.



(a) Results in $\theta \to 2\theta$

(b) Results in Baird seven star counter example

Figure 1: Experiment results

6.1 $\theta \to 2\theta$ (Tsitsiklis & Van Roy, 1996)

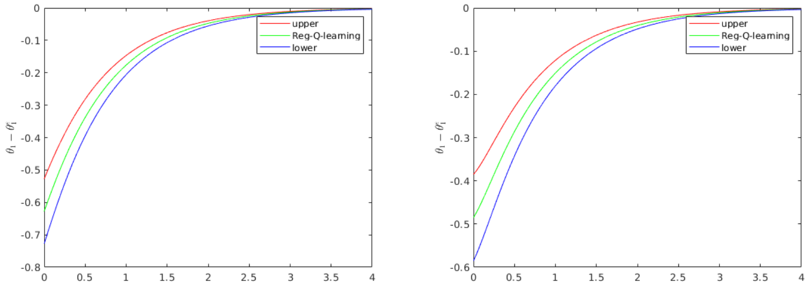
Even when there are only two states, Q-learning with linear function approximation could diverge (Tsitsiklis & Van Roy, 1996). Depicted in Figure 4d in Appendix A.8.1, from state one (θ), the transition is deterministic to absorbing state two (2θ), and reward is zero at every time steps. Therefore, the episode length is fixed to be two. Learning rate for Greedy GQ (GGQ) and Coupled Q Learning (CQL), which have two learning rates, are set as 0.05 and 0.25, respectively as in Carvalho et al., 2020; Maei et al., 2010. Since CQL requires normalized feature values, we scaled the feature value with $\frac{1}{2}$ as in Carvalho et al., 2020, and initialized weights as one. We implemented Q-learning with target network (Zhang et al., 2021), which also have two learning rates, without projection for practical reason (Qtarget). We set the learning rate as 0.25 and 0.05 respectively, and the weight η as two. For RegQ, we set the learning rate as 0.25, and the weight η as two. It is averaged over 100 runs. In Figure 1a, we can see that RegQ achieves the fastest convergence rate.

6.2 Baird Seven Star Counter Example (Baird, 1995)

Baird, 1995 considers an overparameterized example, where Q-learning with linear function approximation diverges. The overall state transition is depicted in Figure 4f given in Appendix A.8.1. There are seven states and two actions for each state, which are solid and dash action. The number of features are $h = 15$. At each episode, it is initialized at random state with uniform probability. Solid action leads to seventh state while dashed action makes transition uniformly random to states other than seventh state. At seventh state, the episode ends with probability $\frac{1}{100}$. The behavior policy selects dashed action with probability $\frac{5}{6}$, and solid action with probability $\frac{1}{6}$. Since CQL in Carvalho et al. (2020) converges under normalized feature values, we scaled the feature matrix with $\frac{1}{\sqrt{5}}$. The weights are set as one except for $\theta_7 = 2$. The learning rates and the weight η are set as same as the previous experiment. As in Figure 1b, Qtarget shows the fastest convergence but to guarantee convergence, it requires projection twice theoretically, which is not implemented in this experiment. Our RegQ shows fastest convergence compared to other two time scale algorithm, CQL and GGQ.

6.3 Mountain Car (Sutton & Barto, 2018)

Mountain Car is environment where state consists of position, and velocity, which are both continuous values. The actions are discrete, accelerating to left, staying neutral, and accelerating to the right. The goal is to reach the top of the mountain quickly as agent gets -1 reward every time step. We use tile-coding (Sutton & Barto, 2018) to discretize the states. We experimented under various tiling numbers and with appropriate η, it achieves performance as Q-learning does. We ran 1000 episodes for the training process, and the episode reward was averaged for 100 runs during test time. From Table 1, with appropriate η, RegQ performs comparable to Q-learning.



(a) $\theta_1 - \theta_1^e$ (b) $\theta_2 - \theta_2^e$

Figure 2: O.D.E. results

Table 1: Mountain Car results, step size = 0.1. The columns correspond to η, and rows correspond to number of tiles.

	0	0.05	0.1	0.2
2×2	-199.803 ± 0.025	-200.0 ± 0.0	-199.767 ± 0.028	$\mathbf{-199.725 \pm 0.032}$
4×4	-191.622 ± 0.216	-191.96 ± 0.204	-189.17 ± 0.241	$\mathbf{-187.943 \pm 0.256}$
8×8	-164.954 ± 0.255	-163.768 ± 0.27	$\mathbf{-162.577 \pm 0.25}$	-163.03 ± 0.255
16×16	-157.95 ± 0.252	$\mathbf{-157.447 \pm 0.245}$	-157.643 ± 0.257	-159.828 ± 0.25

6.4 O.D.E. Experiment

Let us consider a MDP with $|\mathcal{S}| = 2$, $|\mathcal{A}| = 2$, and the following parameters:

$$X = \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 1 & 0 \\ 0 & 2 \end{bmatrix}, \quad D = \begin{bmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{4} \end{bmatrix}, \quad P = \begin{bmatrix} 0.5 & 0.5 \\ 1 & 0 \\ 0.5 & 0.5 \\ 0.25 & 0.75 \end{bmatrix}, \quad R = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \gamma = 0.99.$$

For this MDP, we will illustrate trajectories of the upper and lower system. Each state action pair is sampled uniformly random and reward is one for every time step. $\eta = 2.25$ is chosen to satisfy conditions of Theorem 5.1. From Figure 2, we can see that the trajectory of the original system is bounded by the trajectories of lower and upper system.

7 Conclusion

In this paper, we presented a new convergent Q-learning with linear function approximation (RegQ), which is simple to implement. We provided theoretical analysis on the proposed RegQ, and demonstrated its performance on several experiments, where the original Q-learning with linear function approximation diverges. Developing a new Q-learning algorithm with linear function approximation without bias would be one interesting future research topic. Moreover, considering the great success of deep learning, it would be interesting to develop deep reinforcement learning algorithms with appropriately preconditioned regularization term instead of using the target network.

REFERENCES

Naman Agarwal, Syomantak Chaudhuri, Prateek Jain, Dheeraj Nagaraj, and Praneeth Netrapalli. Online target q-learning with reverse experience replay: Efficiently finding the optimal policy for linear mdps. *arXiv preprint arXiv:2110.08440*, 2021.

Praveen Agarwal, Mohamed Jleli, and Bessem Samet. Fixed point theory in metric spaces. *Recent Advances and Applications*, 2018.

Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, pp. 507–517. PMLR, 2020.

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pp. 30–37. Elsevier, 1995.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279, 2013.

Vivek S Borkar and Sean P Meyn. The ode method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000.

Steven J Bradtke and Andrew G Barto. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22(1):33–57, 1996.

Diogo Carvalho, Francisco S Melo, and Pedro Santos. A new convergent variant of q-learning with linear function approximation. *Advances in Neural Information Processing Systems*, 33:19412–19421, 2020.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model. *arXiv preprint arXiv:2101.05982*, 2021.

Daniela Pucci De Farias and Benjamin Van Roy. On the existence of fixed points for approximate value iteration and temporal-difference learning. *Journal of Optimization theory and Applications*, 105(3):589–608, 2000.

Adithya M Devraj and Sean Meyn. Zap q-learning. *Advances in Neural Information Processing Systems*, 30, 2017.

Raghuram Bharadwaj Diddigi, Chandramouli Kamanchi, and Shalabh Bhatnagar. A convergent off-policy temporal difference algorithm. *arXiv preprint arXiv:1911.05697*, 2019.

Sina Ghiassian, Andrew Patterson, Shivam Garg, Dhawal Gupta, Adam White, and Martha White. Gradient temporal-difference learning with regularized corrections. In *International Conference on Machine Learning*, pp. 3524–3534. PMLR, 2020.

Abhijit Gosavi. Boundedness of iterates in q-learning. *Systems & control letters*, 55(4):347–349, 2006.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Morris W Hirsch and Hal Smith. Monotone dynamical systems. In *Handbook of differential equations: ordinary differential equations*, volume 2, pp. 239–357. Elsevier, 2006.

RA Horn and CR Johnson. Matrix analysis second edition, 2013.

Tommi Jaakkola, Michael I Jordan, and Satinder P Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural computation*, 6(6):1185–1201, 1994.

Hassan K Khalil. *Nonlinear systems; 3rd ed.* Prentice-Hall, Upper Saddle River, NJ, 2002. URL https://cds.cern.ch/record/1173048. The book can be consulted by contacting: PH-AID: Wallet, Lionel.

Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. Maxmin q-learning: Controlling the estimation bias of q-learning. *arXiv preprint arXiv:2002.06487*, 2020.

Donghwan Lee and Niao He. A unified switching system perspective and ode analysis of q-learning algorithms. *arXiv preprint arXiv:1912.02270*, 2019.

Donghwan Lee, Han-Dong Lim, Jihoon Park, and Okyong Choi. Versions of gradient temporal difference learning. *arXiv preprint arXiv:2109.04033*, 2021.

Daniel Liberzon. *Switching in systems and control*. Springer Science & Business Media, 2003.

Hamid Reza Maei. Gradient temporal-difference learning algorithms. 2011.

Hamid Reza Maei, Csaba Szepesvári, Shalabh Bhatnagar, and Richard S Sutton. Toward off-policy learning control with function approximation. In *ICML*, 2010.

Sridhar Mahadevan, Bo Liu, Philip Thomas, Will Dabney, Steve Giguere, Nicholas Jacek, Ian Gemp, and Ji Liu. Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces. *arXiv preprint arXiv:1405.6757*, 2014.

Francisco S Melo. Convergence of q-learning: A simple proof. *Institute Of Systems and Robotics, Tech. Rep*, pp. 1–4, 2001.

Francisco S Melo, Sean P Meyn, and M Isabel Ribeiro. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on Machine learning*, pp. 664–671, 2008.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

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

Richard S Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 993–1000, 2009.

Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 17(1): 2603–2631, 2016.

John N Tsitsiklis and Benjamin Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1):59–94, 1996.

John N Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE transactions on automatic control*, 42(5):674–690, 1997.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International Conference on Machine Learning*, pp. 6995–7004. PMLR, 2019.

Shangtong Zhang, Hengshuai Yao, and Shimon Whiteson. Breaking the deadly triad with a target network. *arXiv preprint arXiv:2101.08862*, 2021.