The goal of Q-learning: estimating the optimal Q-function $Q^*$. Q-learning with linear function approximation: approximate the optimal Q-function from a pre-specified linear sub-space $W = \{ \phi(x) \in \phi \phi \}$.

Main Contributions:
- Algorithm Design: We design a variant of Q-learning with linear function approximation using target network and truncation, which is driven by a single trajectory of Markovian samples.
- Sample Complexity Bounds: We establish an $O(e^{-c \epsilon})$ sample complexity up to a function approximation error.
- Broad Applicability: Our results were established under minimal assumptions, in particular, as long as the behavior policy is sufficient exploitative.

Q-Learning with Linear Function Approximation:

**Target Network and Truncation**

**Algorithm 1** Q-Learning with Linear Function Approximation: Target Network and Truncation

1. Initialize $\theta_0 = 0$ for all $t = 0, 1, \ldots, T$ and $\theta_0 = 0$, behavior policy $\pi_0$.
2. for $t = 0, 1, \ldots, T$ do
3. for $k = 0, 1, \ldots, K$ do
4. Sample $\phi_0 \sim \pi_0(s_t, a_t)$.
5. $\theta_{k+1} \leftarrow \theta_k + \alpha \phi_0(s_t, a_t) \nabla Q(s_t, a_t) + \gamma \max_{\phi_1} \hat{J}(s_{t+1}, a_{t+1}, \theta_{k+1}) - \phi_0(s_t, a_t) \nabla J(s_t, a_t)$.
6. end for
7. $\theta_{k+1} = \theta_k$.
8. $s_{t+1} = s_t$.
9. end for
10. Output: $\theta_T$.

**Target network**: The parameter $\theta_k$ represents the target network, which is fixed in the inner-loop and is synchronized to the last iterate $\theta_k$ in the outer-loop.

**Truncation**: Truncation operator: $\{ \begin{array}{ll} [\min \{\phi(s_t, a_t)\}, \max \{\phi(s_t, a_t)\}] \{ s_t, a_t \} = \\ \{ \phi(s_t, a_t) \} (1 - \gamma) / (1 - \gamma) \\ \{ s_t, a_t \} \leq 1 / (1 - \gamma) \end{array}$

**Finite-Sample Guarantees**

Assumption: The behavior policy $\pi_0$ satisfies $\pi_0(\phi) > 0$ for all $(x, a)$, and induces a uniformly ergodic Markov chain $\{s_t\}$, whose stationary distribution is denoted by $\mu$.

Some Notations:
- $D \in \mathbb{R}^{d \times d}$: a diagonal matrix with diagonal entries $D_{ij} = \phi_j(\phi_i)(\phi_i)$.
- $\lambda_{\min}: \text{the minimum eigenvalue of the matrix } D^T D$.
- $\epsilon_{\text{approx}}$: function approximation error defined by $\epsilon_{\text{approx}} = \max_{s_t, a_t, \theta_k} | \hat{J}(s_t, a_t, \theta_k) - J(s_t, a_t) |$.
- $\mathcal{H}(\cdot)$ is the Bellman optimality operator, and $\mathcal{H}(\cdot)$ denotes the projection operator onto $W$ with respect to a weighted $L_2$-norm.

**Theorem (finite-sample bounds)**: When using properly chosen constant stepsize $\alpha = \gamma$, we have:

$$E[\|Q - Q^*\|^2] \leq \frac{\epsilon_{\text{approx}}^2}{1 - \gamma} + \frac{1 - \lambda_{\min}^2 + \sqrt{\lambda_{\min}^2} + \alpha^2}{(1 - \lambda_{\min})^2} \gamma E[\|Q - Q^*\|^2].$$

**Error Term $E_1$**: Suppose we are using a complete basis and are able to perform fixed-point iteration to solve the Bellman optimality equation. Then $E_1$ is the only error term.

**Error Term $E_2$**: This term arises due to using linear function approximation and vanishes when we have a complete basis.

**Error Term $E_3$**: This term captures the error in the inner-loop, which can be viewed as a stochastic variant of performing the desired fixed-point iteration.

**Corollary (Sample Complexity)**: To achieve $E[\|Q - Q^*\|^2] \leq \epsilon + E_3$, the number of samples required is

$$O \left( \frac{1}{(1 - \gamma)^2} / \epsilon \right).$$

The $e^{-2}$ sample complexity matches with the sample complexity of Q-learning in the tabular setting and is known to be optimal.

**The Reason that Target Network and Truncation Stabilize Q-Learning**

**Summary**

- **Semi-gradient Q-learning**: The algorithm diverges for Baird’s MDP example.
- **Target network Q-learning**: The algorithm is a stochastic variant of projected Bellman iteration, and hence converges for Baird’s MDP example.
- **Insufficiency of target network**: Target network alone is not sufficient to break the deadly triad.
- **Truncation to the Rescue**: Target network and truncation together provably stabilize Q-learning.

**Classical Semi-Gradient Q-Learning**

**The Algorithm**:

$$\theta_{k+1} = \theta_k + \alpha \phi(s_k, a_k) [\mathcal{H}(s_k, a_k) - \mathcal{F}(s_k, a_k)]$$

where $\mathcal{H}(s_k, a_k) = \mathcal{H}(s_k, a_k)$

**Baird’s Divergent Counter-Example**: It is a 2-state 2-action MDP and satisfies $|s(s)| = d$, so it is essentially a change of basis. Semi-gradient Q-learning converging diverges for this example.

The key in the algorithm design of semi-gradient Q-learning: Semi-gradient Q-learning is a stochastic version of the following deterministic iterative algorithm:

$$\theta_{k+1} = \theta_k + \alpha \mathcal{F}(s_k, a_k)$$

which aims at solving the following equation:

$$\mathcal{F}(s_k, a_k) = 0.$$ 

The previous equation is equivalent to the fixed-point equation

$$\mathcal{F}(s_k, a_k) = 0.$$ 

and the corresponding fixed-point iteration is given by

$$\theta_{k+1} = \mathcal{F}(s_k, a_k)$$

For Baird’s MDP example, the green iterative algorithm converges while the blue one (which is the deterministic version of semi-gradient Q-learning) does not.

**Introducing Target Network**

**Motivation**: We need to design a stochastic version of the green iterative algorithm, which will at least converge for Baird’s MDP example.

**Algorithm 2** Q-Learning with Linear Function Approximation: Target Network and No Truncation

1. Input: Images $T$, $K$, initializations $\theta_0 = 0$ for all $t = 0, 1, \ldots, T$ and $\theta_0 = 0$, behavior policy $\pi_0$.
2. for $t = 0, 1, \ldots, T$ do
3. for $k = 0, 1, \ldots, K$ do
4. Sample $\phi_0 \sim \pi_0(s_t, a_t)$.
5. $\theta_{k+1} = \theta_k + \alpha \phi_0(s_t, a_t) \nabla Q(s_t, a_t) + \gamma \max_{\phi_1} \hat{J}(s_{t+1}, a_{t+1}, \theta_{k+1}) - \phi_0(s_t, a_t) \nabla J(s_t, a_t)$.
6. end for
7. $\theta_{k+1} = \theta_k$.
8. $s_{t+1} = s_t$.
9. end for
10. Output: $\theta_T$.

**Why Algorithm 2 is a stochastic version of the Green iteration**: Inner-Loop: Line 5 is a convergent linear stochastic approximation algorithm aims at finding $\theta_N$ that solves $\mathcal{F}(\theta_N) = 0$.

Therefore, we can use $\theta_N$ as initialization for $\theta_k$ when $K$ is large.

Outer-Loop: $\theta_k$ is synchronized to $\theta_N$.

**Together**: Algorithm 2 is a stochastic version of

$$\theta_{k+1} = \mathcal{F}(s_k, a_k)$$

**Numerical Simulation for Baird’s MDP Example**

**Conclusion**

We provide a stable design of Q-learning with linear function approximation using target network and truncation. Moreover, we establish finite sample guarantees (which implies an optimal $O(e^{-c \epsilon})$ sample complexity) up to a function approximation error.

**Future Directions**
- Establishing the asymptotic (almost sure) convergence
- Extending our results to using neural network approximation

---

1. California Institute of Technology, 2. The University of Texas at Austin, 3. Georgia Institute of Technology,