Notes

Paper Summary

Yu Chen

The Chinese University of HongKong, Department of Mechanical and Automation Engineering

E-mail: anschen@link.cuhk.edu.hk

Contents

1 Policy Gradient Methods for Reinforcement Learning with Function Approximation

This article[1] show theoretical possibility of using approximator to encode a policy. In this paper, the author prove the results for both two value functions, one is average reward,

$$V^{\pi}(s) = \lim_{n\to\infty} \frac{1}{n}\mathbb{E}[r_1 + r_2 + \ldots + r_n | s, \pi] \tag{1.1}$$

$$Q^{\pi}(s,a) = \mathbb{E}[\sum_{t=1}^{+\infty} r_t - V^{\pi}(s) | s, a, \pi], \tag{1.2}$$

the other is state-by-state reward,

$$V^{\pi}(s) = \mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r_t | s, \pi], \tag{1.3}$$

$$Q^{\pi}(s,a) = \mathbb{E}[\sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k} | s_t = s, a_t = a, \pi] \tag{1.4}$$

We use parameters θ to approximate policy π, which means $\pi(s,a) = \pi(s,a;\theta)$.

Theorem 1.0.1 (Policy Gradient). *For any MDP, in either the average-reward or state-state formulatios,*

$$\frac{\partial V^{\pi}(s)}{\partial \theta} = \int dsda \rho^{\pi}(s) \frac{\partial \pi(s,a)}{\partial \theta} Q^{\pi}(s,a), \tag{1.5}$$

where $\rho^{\pi}(s)$ is discounted density for state s, defined as,

$$\rho^{\pi}(s) = \lim_{t\to\infty} p(s_t|s_0,\pi), \text{stationary distribution for average reward}, \tag{1.6}$$

$$\rho^{\pi}(s) = \sum_{t=0}^{\infty} \gamma^t p(s_t = s|s_0, \pi), \mathrm{state-state\ formulation}. \tag{1.7}$$

Proof:

Lemma 1.0.1. *The state value function can be written as,*

$$V^{\pi}(s) = \int dsda \rho^{\pi}(s)\pi(s,a)R_s^a, \tag{1.8}$$

where $R_s^a = \int ds' r p(r, s'|s,a)$, where $p(r,s'|s,a)$ is transition probability of MDPs.

Now, we firstly prove average-reward scheme.

$$\begin{aligned}
\frac{\partial V^{\pi}(s)}{\partial \theta} &= \frac{\partial}{\partial \theta}\sum_a \pi(s,a)Q^{\pi}(s,a) && (1.9)\\
&= \sum_a \partial_\theta \pi Q^{\pi} + \pi \partial_\theta Q^{\pi} && (1.10)\\
&= \sum_a \partial_\theta \pi Q^{\pi} + \pi \partial_\theta \left(R_s^a - V^{\pi}(s) + \sum_{s'} p(s'|s,a)V^{\pi}(s') \right) && (1.11)\\
&= \sum_a \left(\partial_\theta \pi Q^{\pi} + \pi \sum_{s'} p(s'|s,a)\partial_\theta V^{\pi}(s') \right) - \partial_\theta V^{\pi}(s). && (1.12)
\end{aligned}$$

Since, for averaged reward, the value function depends on the final stationary distribution, we can,

$$\begin{aligned}\sum_s \rho^\pi(s)\partial_\theta V^\pi(s) =& \sum_{s,a} \rho^\pi(s) Q^\pi(s,a)\partial_\theta \pi(s,a) + \sum_{s,a,s'} p(s'|s,a)\pi(s,a)\rho^\pi(s)\partial_\theta V^\pi(s') \\ & - \sum_s \rho^\pi(s)\partial_\theta V^\pi(s) \\ =& \sum_{s,a} \rho^\pi(s) Q^\pi(s,a)\partial_\theta \pi(s,a) + \sum_{s'} \rho^\pi(s')\partial_\theta V^\pi(s') - \sum_s \rho^\pi(s)\partial_\theta V^\pi(s) \\ =& \sum_{s,a} \rho^\pi(s) Q^\pi(s,a)\partial_\theta \pi(s,a).\end{aligned} \tag{1.13}$$

QED.

For state-state formulation, we can compute as,

$$\partial_\theta V^\pi(s) = \sum_a \partial_\theta \pi Q^\pi + \pi \partial_\theta Q^\pi \tag{1.14}$$

$$= \sum_a \partial_\theta \pi Q^\pi + \pi \partial_\theta \sum_{s'} rp(r,s'|s,a) + \gamma p(s'|s,a)V^\pi(s') \tag{1.15}$$

$$= \sum_a \partial_\theta \pi(s,a) Q^\pi(s,a) + \pi(s,a)\gamma p(s'|s,a)\partial_\theta V^\pi(s') \tag{1.16}$$

$$\ddots \tag{1.17}$$

$$= \sum_s \rho^\pi(s) \sum_a Q^\pi(s,a)\partial_\theta \pi(s,a). \tag{1.18}$$

From the above, we can find gradient of policy depends only on $Q^\pi(s,a)$ and is independent of $\partial_\theta Q^\pi$, which brings greate convinience. Then, we can introduce another approximation with parameters ω to approximate Q value function, denoted by $f_w(s,a)$. To make $f_w \to Q$, the following equation should be satisfied,

$$\begin{aligned}&\partial_w \sum_s \sum_a \rho^\pi(s,a)\pi(s,a)(Q(s,a) - f_w(s,a))^2 = 0 \\ \Leftrightarrow & \sum_s \sum_a \rho^\pi(s,a)\pi(s,a)(Q(s,a) - f_w(s,a))\partial_w f_w(s,a) = 0.\end{aligned} \tag{1.19}$$

Theorem 1.0.2 (Policy Gradient with Function Approximation)**.** *If f_ω satisfies Eq.(1.19) and,*

$$\frac{\partial f_w(s,a)}{\partial \omega} = \frac{\partial \pi(s,a)}{\partial \theta}\frac{1}{\pi(s,a)}, \tag{1.20}$$

then,

$$\frac{\partial V^\pi(s)}{\partial \theta} = \sum_s \rho^s(\pi) \sum_a \frac{\partial \pi(s,a)}{\partial \theta} f_w(s,a). \tag{1.21}$$

The proof is obivous. In the following, we can discuss Eq.(1.20), and construct approximators satisfying it. For example, we use a Gibbs distribution to parameterize $\pi(s,a)$ as,

$$\pi(s,a;\theta) = \frac{e^{\theta^T \phi(s,a)}}{\sum_{a'} e^{\theta^T \phi(s,b)}}. \tag{1.22}$$

And then, we obtain,

$$\frac{1}{\pi(s,a)}\frac{\partial \pi(s,a)}{\partial \theta} = \phi(s,a) - \sum_b \pi(s,b)\phi(s,b) \tag{1.23}$$

Hence, we just need to parameterize $f_w(s,a)$ as,

$$f_w(s,a) = w^T(\phi(s,a) - \sum_b \pi(s,b)\phi(s,b)). \tag{1.24}$$

Theorem 1.0.3 (Policy Iteration with Function Approxmimation)**.** *Let π and f_w be any differential function approximators for the policy and value function respectively that satisfy the compatibility condition1.20 and for which $\max_{\theta,s,a,i,j} |\frac{\partial^2 \pi(s,a)}{\partial\theta_I \partial\theta_j}| < B < \infty$. Let $\{\alpha_k\}_{k=0}^{+\infty}$ be any step-size sequence such that $\lim_{k\to\infty} \alpha_k = 0$ and $\sum_k \alpha_k = \infty$. Then for any MDP with bounded rewards, the sequence $\{V^{\pi_k}\}$, defined by any θ_0, π_k, and,*

$$w_k = w \text{ such that } \sum_s \rho^{\pi_k} \sum_a \pi_k(s,a)(Q^{\pi_k}(s,a) - f_w(s,a))\partial_w f_w(s,a) = 0, \tag{1.25}$$

$$\theta_{k+1} = \theta + \alpha_k \sum_s \rho^{\pi_k}(s) \sum_a f_{w_k}(s,a)\partial_\theta \pi_k(s,a), \tag{1.26}$$

converges such that $\lim_{k\to\infty} \frac{\partial V^{\pi_k}(s)}{\partial\theta} = 0$

2 Eligibility Traces

2.1 On policy

Eligibility traces is a technique to find an intermedia between MC and DP. As we known, DP use one step approximation, i.e. $V_{t+1} = V_t + \Delta V_t$, where $\Delta V_t = G_t^{t+1} - V_t := r_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)$. And MC use the whole epoch information, i.e. $G_t^T = r_{t+1} + \gamma r_{t+2} + \gamma^{T-t} r_{T-t+1}$. Hence, we can define a n step approximation G_t^{t+n} as,

$$G_t^{t+n} = r_{t+1} + \gamma r_{t+2} + \gamma^{n-1} r_{t+n} + \gamma^n V_t(S_{t+n}). \tag{2.1}$$

Or, more generally, we can linearly combine these $G_t^{t+n}, \forall n = 1,2,3\cdots$ with exponentially decayed discount.

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{+\infty} \lambda^{n-1} G_t^{t+n}. \tag{2.2}$$

If the epoch terminates at finite time T, we can use

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} G_t^{t+n} + \lambda^{T-t-1} G_t. \tag{2.3}$$

So far, everything is just theoretically nice, since implementation seems that we need to whole information of epoch for $n = 1,2,3\cdots$. Fortunately, we can use backward view to find a practical algorithm. To this end, people introduced a new varibale–*eligibility trace*, which is defined as,

$$E_t(s) = \gamma\lambda E_{t-1}(s), \forall s \neq S_t, \tag{2.4}$$

$$E_t(S_t) = \gamma\lambda E_{t-1}(S_{t-1}) + 1. \tag{2.5}$$

And TD error is defined as,

$$\delta_t = r_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t), \tag{2.6}$$

which is always one-step. And then we can find ΔV_t is,

$$\Delta V_t = \alpha E_t(S_t)\delta_t. \tag{2.7}$$

The above is for tabular method. If we use function approximation, then we want to minimize the loss $(G_t^\lambda - V(S_t, \theta))^2$, then we can update θ as,

$$\theta \leftarrow \theta + \alpha(G_t^\lambda - V(S_t, \theta))\nabla_\theta V(S_t, \theta). \tag{2.8}$$

Hence, eligibility trace algorithm can be defined as,

$$e \leftarrow e + \nabla_\theta V(S_t, \theta) \tag{2.9}$$

$$\theta \leftarrow \theta + \alpha e \delta_t \tag{2.10}$$

$$e \leftarrow \gamma\lambda e. \tag{2.11}$$

Can we use more information in this whole process? In the above setting, $TD(\lambda)$ is from time t to terminal state T or ∞. Maybe we regard each state S_t we meet as a terminal state, and introduce a new variable $G_{t:t+n}^\lambda$ instead of $G_t^\lambda = G_{t:\infty}^\lambda$ by definition,

$$G_{t:t+n}^\lambda = (1-\lambda)\sum_{k=1}^{n-1} \lambda^{k-1} G_{t:t+k} + \lambda^{n-1} G_{t:t+n}. \tag{2.12}$$

We need to compute the TD of this G,

$$\begin{aligned} G_{t:t+n+1}^\lambda - G_{t:t+n}^\lambda &= (1-\lambda)\lambda^{n-1} G_{t:t+n} + \lambda^n G_{t:t+n+1} - \lambda^{n-1} G_{t:t+n} \\ &= \lambda^n (G_{t:t+n+1} - G_{t:t+n}) \\ &= \lambda^n \gamma^n (r_{t+n+1} + \gamma V(S_{t+n+1}) - V(S_{t+n})) \\ &= \lambda^n \gamma^n \delta_{t+n} \end{aligned} \tag{2.13}$$

We expand the whole process in detail. For example, you stay at time t, with current parameter θ_{t-1}, you can update parameters by,

$$\begin{aligned} &\theta_{t,0} = \theta_0, \\ &\theta_{t,1} = \theta_{t,0} + \alpha(G_{0:t}^\lambda - V(S_1, \theta_{t,0}))\nabla_\theta V(S_1, \theta_{t,0}), \\ &\theta_{t,2} = \theta_{t,1} + \alpha(G_{1:t}^\lambda - V(S_2, \theta_{t,1}))\nabla_\theta V(S_2, \theta_{t,1}), \\ &\cdots, \\ &\theta_{t,t} = \theta_{t,t-1} + \alpha(G_{t-1:t}^\lambda - V(S_t, \theta_{t,t-1}))\nabla_\theta V(S_t, \theta_{t,t-1}), \\ &\theta_t = \theta_{t,t}. \end{aligned} \tag{2.14}$$

After that, we reach time $t+1$, do the similar update,

$$\begin{aligned}
&\theta_{t+1,0} = \theta_0, \\
&\theta_{t+1,1} = \theta_{t+1,0} + \alpha(G^\lambda_{0:t+1} - V(S_1, \theta_{t+1,0}))\nabla_\theta V(S_1, \theta_{t+1,0}), \\
&\theta_{t+1,2} = \theta_{t+1,1} + \alpha(G^\lambda_{1:t+1} - V(S_2, \theta_{t+1,1}))\nabla_\theta V(S_2, \theta_{t+1,1}), \\
&\cdots, \\
&\theta_{t+1,t+1} = \theta_{t+1,t} + \alpha(G^\lambda_{t:t+1} - V(S_{t+1}, \theta_{t+1,t}))\nabla_\theta V(S_t, \theta_{t+1,t}), \\
&\theta_{t+1} = \theta_{t+1,t+1}.
\end{aligned} \tag{2.15}$$

Since we only focus the total update in each step, that is from θ_t to θ_{t+1}. Hence we can define $y_k := \theta_{t+1,k} - \theta_{t,k}$, and the recursive relation of y_k is obvious,

$$\begin{aligned}
y_k &= y_{k-1} + \alpha(\lambda^{t-k+1}\gamma^{t-k+1}\delta_t - V(S_{k-1}, \theta_{t+1,k-1}) + V(S_{k-1}, \theta_{t,k-1}))\nabla_\theta V(S_{k-1}, \theta_{t+1,k-1}) + \alpha(G^\lambda_{k-1,t} - V(S \\
&\approx y_{k-1} + \alpha(\lambda^{t-k+1}\gamma^{t-k+1}\delta_t - \nabla_\theta V(S_{k-1}, \theta_{t,k-1}) y^T_{k-1})\nabla_\theta V(S_{k-1}, \theta_{t,k})
\end{aligned} \tag{2.16}$$

where we have used the first order approximation and ignore the second term since it is the second order error. For simiplicity, we define eligibility trace $e_{k-1} = y_k \lambda^{k-t-1}\gamma^{k-t-1}/\delta_t$, then,

$$e_k = \lambda\gamma e_{k-1} + \alpha\nabla_\theta V(S_k, \theta_{t,k}) - \alpha\lambda\gamma\nabla^T_\theta V(S_k, \theta_{t,k}) e_{k-1}\nabla_\theta V(S_k, \theta_{t,k-1}) \tag{2.17}$$

But we can find e_k depends on $\theta_{t,k-1}$, which is opposite to our wish. We need to do some approximation later. Since our interest is $\theta_{t+1,t+1}$ instead of $\theta_{t+1,t}$, we should compute $\theta_{t+1,t+1}$,

$$\begin{aligned}
\theta_{t+1,t+1} &= \theta_{t+1,t} + \alpha(G^\lambda_{t,t+1} - V(S_t, \theta_{t+1,t}))\nabla_\theta V(S_t, \theta_{t+1,t}) \\
&= y_t + \theta_t + \alpha(G^\lambda_{t,t+1} - V(S_t, \theta_{t+1,t}))\nabla_\theta V(S_t, \theta_{t+1,t}) \\
&= \theta_t + \lambda\gamma e_{t-1}\delta_t + \alpha(G^\lambda_{t,t+1} - V(S_t, \theta_{t+1,t}))\nabla_\theta V(S_t, \theta_{t+1,t})
\end{aligned} \tag{2.18}$$

By using the facts,

$$\begin{aligned}
G^\lambda_{t,t+1} - V(S_t, \theta_{t+1,t}) &= \delta_t + V(S_t, \theta_{t-1}) - V(S_t, \theta_{t+1,t}) \\
&= \delta_t - (V(S_t, \theta_{t+1,t}) - V(S_t, \theta_t) + V(S_t, \theta_t) - V(S_t, \theta_{t-1})) \\
&= \delta_t - \nabla_\theta V(S_t, \overline{\theta}_{t+1,t})\lambda\gamma e_{t-1}\delta_t + V(S_t, \theta_t) - V(S_t, \theta_{t-1}).
\end{aligned} \tag{2.19}$$

Combining these two with Eq.(2.17), we can obtain,

$$\theta_{t+1} = \theta_t + e_t\delta_t + \alpha(V(S_t, \theta_{t-1}) - V(S_t, \theta_t))\nabla V(S_t, \theta_{t-1}) \tag{2.20}$$

Repalcing above $\theta_{t,k}$ with information we have, we obtain the following algorithm:

```
On-line eligibility trace with approximator Initialise $e = 0, S, \theta$, and $V_{prev} = V(S, \theta)$
Reapeat:
    Choose action, make transition $\rightarrow (S', R)$
    $V_{curr} = V(S', \theta)$,
    $\delta = \gamma V_{curr} + R - V_{prev}$
    $e = \lambda\gamma e + \alpha\nabla_\theta V(S, \theta) - \alpha\lambda\gamma\nabla_\theta^T V(S, \theta) e \nabla_\theta V(S, \theta)$,
    $\theta = \theta + \delta e + \alpha(v_{prev} - V(S, \theta))\nabla_\theta V(S, \theta)$
    $v_{prev} = v_{curr}, S = S'$
```

2.2 Off policy

3 GTD–gradient temporal difference

3.1 Policy gradient methods for reinforcement learning with function approximation

The article[2] firstly raises idea for off-policy gradient temporal difference with a linear function approximator. We denote the stationary distribution of behavior policy b as $\rho^b(s)$. The observation of state s is given by $\phi_s \in \mathbb{R}^n$, so the δ_t is,

$$\delta_t = r_t + \gamma V_t(s_{t+1}) - V_t(s_t) = r_t + \gamma\theta^T\phi_{t+1} - \theta^T\phi_t. \tag{3.1}$$

Intutively, we wish the δ_t approaches to 0, when we reach a local minimum or maximum. In other words, we can define our obejective function as $L2$ norm,

$$J(\theta) = \mathbb{E}[\delta\phi]^T\mathbb{E}[\delta\phi], \tag{3.2}$$

where $\mathbb{E}$ is over the stationary distribution $\rho^b(s)$. The gradient is,

$$\begin{aligned} \nabla_\theta J(\theta) &= 2\nabla_\theta\mathbb{E}[\delta\phi^T]\mathbb{E}[\delta\phi] && (3.3)\\ &= -2\mathbb{E}[(\phi - \gamma\phi')\phi^T]\mathbb{E}[\delta\phi]. && (3.4) \end{aligned}$$

where $\phi' = \phi_{t+1}$. Hence, there are two approaches two find this gradient. Firstly, it stores the expectation of the first term and samples the second term; secondly, it stores the expectation of the second term and samples the first term. (Note: why we do not stores both expectation is that it will introduce bias brought by the correlation of these two terms). Mathematically, we have $\phi_1, r_1, \phi_2, r_2, \cdots, \phi_k, r_k, \phi_{k+1}, r_{k+1}$,

$$\begin{aligned} A_k &= \frac{1}{k}\sum_{j=1}^{k}(\phi_j - \gamma\phi_{j+1})\phi_j^T, && (3.5)\\ \theta_{k+1} &= \theta_k + \alpha_k A_k \delta_k \phi_k, && (3.6) \end{aligned}$$

where $\delta_k = r_k + \gamma\theta_k^T\phi_{k+1} - \theta_k^T\phi_k$. The disadvantage of this method is that finding $A_k \in \mathbb{R}^{n\times n}$ costs our n^2 complexity. Hence, the most case we use the second approach,

$$\begin{aligned} w_{k+1} &= w_k + \beta_k(\delta_k\phi_k - w_k), && (3.7)\\ \theta_{k+1} &= \theta_k + \alpha_k(\phi_k - \gamma\phi_{k+1})\phi_k^T w_k && (3.8) \end{aligned}$$

3.2 Fast Gradient-Descent Methods for Temporal-Difference Learning with Linear Function Approximation

This article [3] give an improvement of the above algorithm, named GTD2 and TDC(TD with gradient correction). GTD2 uses a different obejcetive function,

$$J(\theta) = \mathbb{E}[\delta\phi]^T \mathbb{E}[\phi\phi^T]^{-1} \mathbb{E}[\delta\phi]. \tag{3.9}$$

The idea of this objective function comes from the following facts. The ultimate goal of these algorithm is to find an approximator V_θ, which can approximates V as exactly as possible. As we known, the true V satisfies the Bellman equation,

$$V = TV, \tag{3.10}$$

where T is Bellman transition matrix. Hence, we hope our approximator can satisfies this property too, in other wrods, we wish $\|V_\theta - TV_\theta\|^2_{\rho^b} \equiv \int ds \rho^b(s)|V_\theta(s) - TV_\theta(s)|^2 \to 0$. Unfornuately, TV_θ may not stay in the space V_θ. Instead, we denote a projector $\mathit{\Pi}$, which projects any v to space V_θ as $\mathit{\Pi} v$. To find $\mathit{\Pi}$, we firstly find θ minimizing $\|v - \Phi\theta\|_D$, where $D \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ is diagonal matrix with element $\rho^b(s)$, and Φ with rows ϕ_s.

$$\nabla_\theta \|v - \Phi\theta\|^2_D = -2\Phi^T D(v - \Phi\theta), \tag{3.11}$$

which means $\theta = (\Phi^T D \Phi)^{-1}\Phi^T D v$, and we know $\mathit{\Pi} v = \Phi\theta, \forall v$. Hence,

$$\mathit{\Pi} = \Phi(\Phi^T D\Phi)^{-1}\Phi^T D \tag{3.12}$$

We define mean-squared projected Bellman error as,

$$\text{MSPBE}(\theta) \equiv \|V_\theta - \mathit{\Pi} T V_\theta\|^2_D. \tag{3.13}$$

By using the facts,

$$\begin{aligned}
&\mathbb{E}[\phi\phi^T] = \sum_s d_s \phi_s \phi_s^T = \Phi^T D \Phi, \\
&\mathbb{E}[\delta\phi] = \sum_s d_s \phi_s (R_s + \gamma \sum_{s'} p_{s,s'} V_\theta(s') - V_\theta(s)) = \Phi^T D(TV_\theta - V_\theta), \\
&\mathit{\Pi}^T D \mathit{\Pi} = D^T \Phi(\Phi^T D \Phi)^{-1} \Phi^T D.
\end{aligned} \tag{3.14}$$

we can simplify MSPBE(θ) as

$$\begin{aligned}
\text{MSPBE}(\theta) &= \|V_\theta - \mathit{\Pi} T V_\theta\|^2_D \\
&= (V_\theta - \mathit{\Pi} T V_\theta)^T D (V_\theta - \mathit{\Pi} T V_\theta) \\
&= (V_\theta - TV_\theta)^T \mathit{\Pi}^T D \mathit{\Pi} (V_\theta - TV_\theta) \\
&= (V_\theta - TV_\theta)^T D^T \Phi(\Phi^T D\Phi)^{-1}\Phi^T D(V_\theta - TV_\theta) \\
&= \mathbb{E}[\delta\phi]^T \mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi].
\end{aligned} \tag{3.15}$$

The gradient is,

$$\nabla_\theta \text{MSPBE}(\theta) = -2\mathbb{E}[(\phi - \gamma\phi')\phi^T]\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi]. \tag{3.16}$$

We use w to approximate $\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi]$, so the update rule is,

$$w_{k+1} = w_k + \beta_k(\delta_k - \phi_k^T w_k)\phi_k, \tag{3.17}$$
$$\theta_{k+1} = \theta_k + \alpha_k(\phi_k - \gamma\phi_k')(\phi_k^T w_k). \tag{3.18}$$

TDC apprixmates this gradient more exactly,

$$\begin{aligned} -\frac{1}{2}\nabla_\theta \mathrm{MSPBE}(\theta) &= \mathbb{E}[(\phi - \gamma\phi')\phi^T]\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi] && (3.19)\\ &= \mathbb{E}[\delta\phi] - \gamma\mathbb{E}[\phi'\phi^T]\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi] && (3.20)\\ &\approx \mathbb{E}[\delta\phi] - \gamma\mathbb{E}[\phi'\phi^T]w && (3.21)\end{aligned}$$

Hence, TDC update θ as,

$$\theta_{k+1} = \theta_k + \alpha_k\delta_k\phi_k - \alpha\gamma\phi_k'(\phi_k^T w_k). \tag{3.22}$$

Gradient Temporal Diffierence

$$w_{k+1} = w_k + \beta_k(\delta_k - \phi_k^T w_k)\phi_k, \tag{3.23}$$
$$\theta_{k+1} = \theta_k + \alpha_k(\phi_k - \gamma\phi_k')(\phi_k^T w_k) \quad \textbf{(GTD2)}, \tag{3.24}$$
$$\theta_{k+1} = \theta_k + \alpha_k\delta_k\phi_k - \alpha\gamma\phi_k'(\phi_k^T w_k) \quad \textbf{(TDC)}. \tag{3.25}$$

3.3 GTD(λ)

Above two algorithms are about GTD(0), in these section, the author introduce the final version of GTD–GTD(λ)[4]. Firstly, we can write $\lambda-$return as,

$$G_t^\lambda = r_{t+1} + \gamma\left((1-\lambda)V(s_{t+1}) + \lambda G_{t+1}^\lambda\right). \tag{3.26}$$

Obviously, when $\lambda = 1$ it is MC method, $\lambda = 0$ is dynamical programming. For simiplicity, we also use linear approximation, so the δ return is,

$$G_t^\lambda = r_{t+1} + \gamma(1-\lambda)\theta^T\phi_{t+1} + \gamma\lambda G_{t+1}^\lambda, \tag{3.27}$$
$$\delta_t^\lambda = G_t^\lambda - \theta^T\phi_t \tag{3.28}$$

Similarly, we can define a new operator as,

$$\mathcal{P}_b^\pi \delta_t^\lambda \phi_t := \sum_s \rho^b(s)\mathbb{E}[\delta_t^\lambda | S_t = s, \pi]\phi(s) \tag{3.29}$$

We can find it has the following properties,

$$\mathbb{E}[\delta_t^\lambda | S_t = s, \pi] = \mathbb{E}[G_t^\lambda - \theta^T\phi_s | S_t = s, \pi] = TV_\theta(s) - V_\theta(s) \tag{3.30}$$

then,

$$\mathcal{P}_b^\pi \delta_t^\lambda \phi_t = \sum_s \rho^b(s)\mathbb{E}[\delta_t^\lambda | S_t = s, \pi]\phi(s) \tag{3.31}$$

$$= \sum_s \rho^b(s)(TV_\theta(s) - V_\theta(s))\phi(s) \tag{3.32}$$

$$= \Phi^T D(TV_\theta - V_\theta). \tag{3.33}$$

Hence,

$$\text{MSPBE}(\theta) = \|V_\theta - \mathit{\Pi} TV_\theta\|_D^2 = (\mathcal{P}_b^\pi \delta_t^\lambda \phi_t)^T \mathbb{E}[\phi_t \phi_t^T]^{-1} \mathcal{P}_b^\pi \delta_t^\lambda \phi_t. \tag{3.34}$$

Here there is a problem $\mathcal{P}_b^\pi \delta_t^\lambda \phi_t$ is along trajectory by π, but our sample is along b. Hence we need to use importance sampling to solve it. To this end, we define,

$$G_t^{\lambda\rho}(\theta) = \rho_t(r_{t+1} + \gamma((1-\lambda)\theta^T \phi_t + \lambda G_t^{\lambda\rho}(\theta))), \tag{3.35}$$

$$\delta_t^{\lambda\rho}(\theta) = G_t^{\lambda\rho}(\theta) - \theta^T \phi_t. \tag{3.36}$$

where $\rho_t = \frac{\pi(S,A)}{b(S,A)}$.

Theorem 3.3.1 (Off-policy with importance sampling)**.**

$$\mathcal{P}_b^\pi \delta_t^\lambda(\theta)\phi_t = \mathbb{E}[\delta_t^{\lambda\rho}(\theta)\phi_t]. \tag{3.37}$$

Proof: To prove this, we only need to prove the following,

$$\mathbb{E}[G_t^{\lambda\rho}\phi_t | S_t = s] = \mathbb{E}[G_t^\lambda \phi_t | S_t = s, \pi]. \tag{3.38}$$

We compute the left side,

$$\begin{aligned}\mathbb{E}[G_t^{\lambda\rho} | S_t = s] =& \mathbb{E}[\rho_t R_{t+1} + \rho_t \gamma(1-\lambda)\theta^T \phi_t | S_t = s] + \gamma\lambda \mathbb{E}[\rho_t G_{t+1}^{\lambda\rho} | S_t = s] \\ =& \mathbb{E}[R_{t+1} + \gamma(1-\lambda)\theta^T \phi_t | S_t = s, \pi] + \\ &+ \gamma\lambda \sum_{s'} p(s'|s,a) b(s,a) \frac{\pi(s,a)}{b(s,a)} \mathbb{E}[G_{t+1}^\lambda | S_{t+1} = s'] \\ =& \mathbb{E}[R_{t+1} + \gamma(1-\lambda)\theta^T \phi_t | S_t = s, \pi] + \lambda\gamma \mathbb{E}[G_{t+1}^\lambda | S_t = s, \pi] \\ =& \mathbb{E}[G_t^\lambda | S_t = s, \pi].\end{aligned} \tag{3.39}$$

Hence,

$$\text{MSPBE}(\theta) = \mathbb{E}[\delta_t^{\lambda\rho}\phi_t]^T \mathbb{E}[\phi_t \phi_t^T]^{-1} \mathbb{E}[\delta_t^{\lambda\rho}\phi_t]. \tag{3.40}$$

This is forward view, to implement this algorithm, we need to find e_t such that $\mathbb{E}[\delta_t^{\lambda\rho}\phi_t] = \mathbb{E}[\delta_t e_t]$, where δ_t is common TD difference $\delta_t = r_{t+1} + \gamma\theta^T \phi_{t+1} - \theta^T \phi_t$.

Theorem 3.3.2 (Equivalence of the TD forward-view and backward-view)**.** *The forward-view is equivalent to the following backward-view:*

$$\mathbb{E}[\delta_t^{\lambda\rho}\phi_t] = \mathbb{E}[\delta_t e_t] \tag{3.41}$$

with updating rule,

$$e_t = \rho_t(\phi_t + \gamma\lambda e_{t-1}). \tag{3.42}$$

Proof: We simplify $\delta_t^{\lambda\rho} = G_t^{\lambda\rho} - \theta^T\phi_t$ step by step.

$$\begin{aligned} G_t^{\lambda\rho} &= \rho_t(r_{t+1} + \gamma\theta^T\phi_{t+1} - \theta^T\phi_t) + \rho_t\theta^T\phi_t - \lambda\gamma\rho_t\theta^T\phi_{t+1} + \lambda\gamma\rho_t G_{t+1}^{\lambda\rho} && (3.43)\\ &= \rho_t\delta_t(\theta) + \rho_t\theta^T\phi_t + \lambda\gamma\rho_t\delta_{t+1}^{\lambda\rho}. && (3.44) \end{aligned}$$

Hence,

$$\delta_t^{\lambda\rho} = \rho_t\delta_t + (\rho_t - 1)\theta^T\phi_t + \lambda\gamma\rho_t\delta_{t+1}^{\lambda\rho}. \tag{3.45}$$

We firstly prove the second term vanishes under expectation,

$$\begin{aligned} \mathbb{E}[(1-\rho_t)\theta^T\phi_t\phi_t] &= \sum_{s,a} b(s,a)\rho^b(s)(1 - \frac{\pi(s,a)}{b(s,a)})\theta^T\phi_s\phi_s && (3.46)\\ &= \sum_{s,a} \rho^b(s)(b(s,a) - \pi(s,a))\theta^T\phi_s\phi_s && (3.47)\\ &= \sum_{s} \rho^b(s)(1-1)\theta^T\phi_s\phi_s = 0. && (3.48) \end{aligned}$$

Hence,

$$\begin{aligned} \mathbb{E}[\delta_t^{\lambda\rho}\phi_t] &= \mathbb{E}[\rho_t\delta_t\phi_t + \lambda\gamma\rho_t\delta_{t+1}^{\lambda\rho}\phi_t]\\ &= \mathbb{E}[\rho_t\delta_t\phi_t] + \lambda\gamma\mathbb{E}[\rho_{t-1}\delta_t^{\lambda\rho}\phi_{t-1}]\\ &= \mathbb{E}[\rho_t\delta_t\phi_t + \lambda\gamma\rho_{t-1}\phi_{t-1}\rho_t\phi_t] + \gamma^2\lambda^2\mathbb{E}[\rho_{t-1}\rho_t\delta_{t+1}^{\lambda\rho}\phi_{t-1}\phi_t]\\ &\ddots\\ &= \mathbb{E}[\delta_t\rho_t(\phi_t + \lambda\gamma\phi_{t-1}\rho_{t-1} + \cdots)]\\ &= \mathbb{E}[\delta_t e_t], \end{aligned} \tag{3.49}$$

with $e_t = \rho_t(\phi_t + \gamma\lambda e_{t-1})$.
Now, we can find gradient of $\text{MSPBE}(\theta) = \mathbb{E}[\delta_t e_t]^T\mathbb{E}[\phi_t\phi_t^T]^{-1}\mathbb{E}[\delta_t e_t]$.

$$-\frac{1}{2}\nabla_\theta\text{MSPBE}(\theta) = -\mathbb{E}[\gamma(\phi_{t+1} - \phi_t)e_t^T]\mathbb{E}[\phi_t\phi_t^T]^{-1}\mathbb{E}[\delta_t e_t]. \tag{3.50}$$

To simplify it, we use the following equalities,

$$\mathbb{E}[\phi_t\rho_t\phi_t^T] = \mathbb{E}[\phi_t\phi_t^T], \mathbb{E}[\phi_{t+1}\rho_t e_t^T] = \mathbb{E}[\phi_{t+1}e_t^T] \tag{3.51}$$

$$\begin{aligned} \mathbb{E}[\phi_t e_t^T] &= \mathbb{E}[\phi_t\rho_t(\phi_t^T + \lambda\gamma e_{t-1}^T)] && (3.52)\\ &= \mathbb{E}[\phi_t\phi_t^T] + \gamma\lambda\mathbb{E}[\phi_{t+1}e_t^T]. && (3.53) \end{aligned}$$

$$\begin{aligned} -\frac{1}{2}\nabla_\theta\text{MSPBE}(\theta) &= \mathbb{E}[\delta_t e_t] - \gamma(1-\lambda)\mathbb{E}[\phi_{t+1}e_t^T]\mathbb{E}[\phi_t\phi_t^T]^{-1}\mathbb{E}[\delta_t e_t]\\ &\approx \mathbb{E}[\delta_t e_t] - \gamma(1-\lambda)\mathbb{E}[\phi_{t+1}e_t^T]w. \end{aligned} \tag{3.54}$$

Hence updating rule is,

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha_t(\delta_t e_t - \gamma(1-\lambda)(e_t^T w_t)\phi_{t+1}), && (3.55)\\ w_{t+1} &= w_t + \beta_t(\delta_t e_t - (w_t^T\phi_t)\phi_t), && (3.56)\\ e_{t+1} &= \rho_t(\phi_t + \gamma\lambda e_{t-1}). && (3.57) \end{aligned}$$

GTD(λ) with linear function approximation

```
Initialize $w_0$ to 0, and $\theta_0$ arbitarily.
Choose $\alpha, \beta, \gamma, \lambda$
Reapeat for each episode:
    Initilize $e = 0$
    Take $A_t$ from $S_t$ according to $b(a|s)$ and observe $(\phi_t, r_{t+1}, \phi_{t+1})$
    Update as:
        $\delta_t \leftarrow r_{t+1} + \gamma\theta^T\phi_{t+1} - \theta^T\phi_t.$
        $\rho_t \leftarrow \frac{\pi(a_t|s_t)}{b(a_t|s_t)}$
        $e_t \leftarrow \rho_t(\phi_t + \lambda\gamma e_{t-1})$
        $\theta_{t+1} \leftarrow \theta_t + \alpha_t[\delta_t e_t - \gamma(1-\lambda)(e_t^T w_t)\phi_{t+1}]$
        $w_{t+1} \leftarrow w_t + \beta_t(\delta_t e_t - (\phi_t^T w_t)\phi_t).$
```

4 Off Policy Actor-Critic

This article[5] introduce the first actor-critic method that can be applied off-policy, give off-policy gradient theorem and convergence proof. In this paper, author use b to denote the behaviour policy, and stationary distribution or discounted distribution is $\rho^b(s)$ in our notations. The objective function is,

$$J(\pi_\theta) = \int \mathrm{d}s\rho^b(s)V^{\pi_\theta}(s). \tag{4.1}$$

Obviously, we should updata θ along the direction $\nabla_\theta J(\pi_\theta)$, which is given by,

$$\begin{aligned}\nabla_\theta J(\pi_\theta) &= \int \mathrm{d}s\mathrm{d}a\rho^b(s)\nabla_\theta(\pi_\theta(s,a)Q^{\pi_\theta}(s,a)) \\ &= \int \mathrm{d}s\mathrm{d}a\rho^b(s)\left(\nabla\pi_\theta Q + \pi_\theta\nabla_\theta Q\right) \\ &\equiv g(\theta) + \int \mathrm{d}s\mathrm{d}a\rho^b(s)\pi_\theta\nabla_\theta Q.\end{aligned} \tag{4.2}$$

Actually, the second term is very difficult to compute, the following theorem proves that ignoring it is good approximation.

Theorem 4.0.1 (Policy Improvement)**.** *Given any policy with parameters θ, let,*

$$\theta' = \theta + \alpha g(\theta). \tag{4.3}$$

Then there exists an $\varepsilon > 0$ such that, for all possible $\alpha < \varepsilon$,

$$J(\pi_{\theta'}) > J(\pi_\theta). \tag{4.4}$$

Further, if π has a tabular representation, then

$$V^{\pi_{\theta'}}(s) > V^{\pi_\theta}(s), \forall s. \tag{4.5}$$

Proof: It is easy to show,

$$\begin{aligned} J(\pi_\theta) &= \int \mathrm{d}s\mathrm{d}a\rho^b(s)\pi_\theta(s,a)Q^{\pi_\theta}(s,a) \\ &\le \int \mathrm{d}s\mathrm{d}a\rho^b(s)\pi_{\theta'}(s,a)Q^{\pi_\theta}(s,a). \end{aligned} \tag{4.6}$$

Since we have $f(x+\alpha\nabla f) - f(x) = \alpha(\nabla f)^2 > 0$. Then we repeat applying the above inequality, we can find,

$$J(\pi_\theta) \le \mathbb{E}_{s\sim\rho^b, a\sim\pi_{\theta'}} Q^{\pi_\theta}(s,a) \tag{4.7}$$

$$= \mathbb{E}_{s\sim\rho^b}\mathbb{E}_{a\sim\pi_{\theta'}}[r_{t+1} + \gamma V^{\pi_\theta}(s_{t+1})] \tag{4.8}$$

$$\le \mathbb{E}_{s\sim\rho^b}\mathbb{E}_{a\sim\pi_{\theta'}}[r_{t+1} + \gamma r_{t+2} + \gamma V^{\pi_\theta(s_{t+2})}] \tag{4.9}$$

$$\ddots \tag{4.10}$$

$$= J(\pi_{\theta'}). \tag{4.11}$$

The second part is similar since

$$V^{\pi_\theta} = \int \mathrm{d}a\pi_\theta(s,a)Q^{\pi_\theta}(s,a) \le \int \mathrm{d}a\pi_{\theta'}(s,a)Q^{\pi}(s,a). \tag{4.12}$$

By repeating the above formula, we can prove the second part.

Theorem 4.0.2 (Off Policy Gradient Theorem)**.** *Let* $\mathbb{Z} = \{\theta | \nabla_\theta J(\pi_\theta) = 0\}$ *and* $\widetilde{\mathbb{Z}} = \{\theta | g(\theta) = 0\}$*, which are both non-empty by some assumptions. If the value function can be represented by our function class, then,*

$$\mathbb{Z} \in \widetilde{\mathbb{Z}}. \tag{4.13}$$

Moreover, if we use a tabular representation for policy π*, then*

$$\mathbb{Z} = \widetilde{\mathbb{Z}}. \tag{4.14}$$

Proof: This first part is very trivial because of off policy improvement theorem. More specificly, any θ^* such that $\nabla_\theta J(\pi_\theta)|_{\theta^*} = 0$ must means $g(\theta^*) = 0$, otherwise there exists a direction $\frac{g(\theta^*)}{\|g(\theta^*)\|}$ such that $J(\theta)$ increases, which is contradict to $\nabla_\theta J(\theta)|_{\theta^*} = 0$.
For the second part, we have a tabular representation, which means each weight corresponds to exactly one state. Without loss of generality, weights $\theta_{s,1}, \theta_{s,2}, \dots, \theta_{s,s_n}$ are relative to state s. Since $g(\theta^*) = 0$,

$$\int \mathrm{d}s'\mathrm{d}a\rho^b(s)\partial_{\theta_{s,j}}\pi_\theta(s',a)Q^{\pi_\theta}(s',a) = \rho^b(s)\partial_{\theta_{s,j}}\pi_\theta(s,a)Q^{\pi_\theta}(s,a) = 0, \forall j = 1,\dots,s_n. \tag{4.15}$$

If there exists k such that,

$$\int \mathrm{d}s'\mathrm{d}a\pi_\theta(s',a)\partial_{\theta_{s,k}}Q^{\pi_\theta}(s',a) \ne 0, \tag{4.16}$$

which means $J(\pi_\theta)$ will increase for local increasement of $Q^{\pi_\theta}(s,a)$. In other words, we can change our policy a litte to increase our objective function, which means,

$$\sum_{j=1}^{n_s} \int \mathrm{d}s' \mathrm{d}a \partial_{\theta_{s,j}} \pi_\theta(s',a) Q^{\pi_\theta}(s',a) \neq 0. \tag{4.17}$$

Contracted to Eq.(4.15).
Then, we can rewrite $g(\theta)$ as,

$$\begin{aligned} g(\theta) &= \int \mathrm{d}s \rho^b(s) Q^{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a|s) &(4.18)\\ &= \int \mathrm{d}s \rho^b(s) b(a|s) \rho(a|s) Q^{\pi_\theta}(s,a) \nabla \log \pi_\theta(a|s) &(4.19)\\ &= \mathbb{E}[\rho(s,a)\psi(s,a)Q^{\pi_\theta}(s,a)]. &(4.20)\end{aligned}$$

There is a famous result that an arbitrary function of state can be introduced into these equations as a baseline without changing the expected value. Hence, we can write $g(\theta)$ as,

$$g(\theta) = \mathbb{E}[\rho(s,a)\psi(s,a)(Q^\pi(s,a) - V(s))] \approx \mathbb{E}[\rho(s,a)\psi(s,a)(R_t^\lambda - V(s))]. \tag{4.21}$$

where R_t^λ is off-policy $\lambda-$return, defined as,

$$R_t^\lambda = r_{t+1} + \gamma(1-\lambda)V(s_{t+1}) + \gamma\lambda\rho_t R_{t+1}^\lambda \tag{4.22}$$

To find a backward-view implement, we need to find e_t such that,

$$\mathbb{E}[\rho(s,a)\psi(s,a)(R_t^\lambda - V(s))] = \mathbb{E}[\delta_t e_t], \tag{4.23}$$

where $\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$. As shown in paper, updating rule of e_t is given by,

$$e_t = \rho_t(\psi_t + \lambda\gamma e_{t-1}). \tag{4.24}$$

```
Off-PAC algorithm
Initialize the vectors $e_v, e_u$ and $w$ to zero
Initialize the vectors $v, u$ arbitrarily
Initialize the state $s$
For each step:
    Choose an action $a$, according to $b(a|s)$
    Observer resultant reward $r$, and next state $s'$,
        $\delta \leftarrow r + \gamma v^T \phi_{s'} - v^T \phi_s$
        $\rho \leftarrow \pi_u(a|s)/b(a|s)$
    Update the critic (GTD($\lambda$) algorithm):
        $e_v \leftarrow \rho(\phi_s + \gamma\lambda e_v)$
        $v \leftarrow v + \alpha_v[\delta e_v - \gamma(1-\lambda)(w^T e_v)\phi_s]$
        $w \leftarrow w + \alpha_w[\delta e_v - (w^T \phi_s)\phi_s]$
    Update the actor:
        $e_u \leftarrow \rho[\frac{\nabla_u \pi_u(a|s)}{\pi_u(a|s)} + \gamma\lambda e_u]$
        $u \leftarrow u + \alpha_u \delta e_u$
    $s \leftarrow s'$
```

5 DDPG

5.1 Determinisitic Policy Gradient Algorithms

In the previous paper, we have introduced policy gradient methods for stochastic policy. The author in this paper[6] show that the determinsitic policy gradient does indeed exist, and furthermore it has a simple model-free form that simply follows the gradient of the action-value function. In addition, they show that the deterministic policy gradient is the limiting case, as policy varinance trends to zero. of the stochastic policy gradient. In this paper, we use $\mu_\theta(s)$ denote a determinsitic policy, which means when we meet state s, we take action $\mu_\theta(s)$.

Theorem 5.1.1 (Deterministic Policy Gradient Theorem)**.** *Suppose that the MDP satisfies some conditions, these imply that $\nabla\mu_\theta$ and $\nabla_a Q^\mu(s,a)$ exist and that the determinsitic policy gradient exists. Then*

$$\nabla_\theta V^{\mu_\theta} = \int \mathrm{d}s \rho^\mu(s) \nabla_\theta \mu_\theta(s) \nabla Q^\mu(s,a)|_{a=\mu(s)}. \tag{5.1}$$

Proof:

$$\begin{aligned}
\nabla_\theta V^{\mu_\theta} &= \nabla_\theta Q^\mu(s, \mu(s)) && (5.2)\\
&= \nabla_\theta \left(R_{s,\mu(s)} + \gamma \int \mathrm{d}s' p(s'|s, \mu(s)) V^\mu(s') \right) && (5.3)\\
&= \nabla_\theta \mu \nabla_a R_{s,a} + \gamma \int \mathrm{d}s' \nabla_\theta \mu \nabla_a p(s'|s, a) V^\mu(s') + p(s'|s, a) \nabla_\theta V^\mu(s') && (5.4)\\
&= \nabla_\theta \mu \nabla_a \left(R_{s,a} + \gamma \int \mathrm{d}s' p(s'|s, a) V^\mu('s) \right) + \gamma \int \mathrm{d}s' p(s'|s, a) \nabla_\theta V^\mu(s') && (5.5)\\
&= \nabla_\theta \mu \nabla_a Q^\mu(s, a) + \gamma \int \mathrm{d}s' p(s'|s, a) \nabla_\theta V^\mu(s') & \ddots\ & (5.6)\\
&= \int \mathrm{d}s \rho^\mu(s) \nabla_\theta \mu \nabla_a Q^\mu(s, a) && (5.7)
\end{aligned}$$

Theorem 5.1.2. *Consider a stochastic policy $\pi_{\mu_\theta,\sigma}$ such that $\pi_{\mu_\theta,\sigma}(a|s) = \nu_\sigma(\mu_\theta(s), \sigma)$, where σ controls variance and ν_σ satisfies some conditions, then*

$$\lim_{\sigma \downarrow 0} \nabla_\theta V^{\pi_{\mu_\theta,\sigma}} = \nabla_\theta V^{\mu_\theta} \tag{5.8}$$

Instead of author's proof, I use a more abstract notation to show the same thing. We can write the deterministic policy μ_θ as the stochastic policy with density $\pi(a|s) = \delta(\mu(s) - a)$. Using the stochastic policy gradient theorem, we can have,

$$\begin{aligned}
\nabla_\theta V^{\mu_\theta} &= \int \mathrm{d}s \mathrm{d}a \rho^\mu(s) \nabla_\theta \delta(\mu_\theta(s) - a) Q^\mu(s, a) && (5.9)\\
&= -\int \mathrm{d}s \mathrm{d}a \rho^\mu(s) \nabla_\theta \mu \nabla_a \delta(a - \mu(s)) Q^\mu(s, a) && (5.10)\\
&= \int \mathrm{d}s \rho^\mu(s) \nabla_\theta \mu \nabla_a Q^\mu(s, \mu(s)), && (5.11)
\end{aligned}$$

where we use integral by parts to obtain the last equality. To implement, we use Q^w to approximate Q^μ with parameters w. If we use the on-policy to choose action, then updating rule is,

Algorithm 5.1.1 (OPDAC(on-policy deterministic actor-critic))**.**

$$\begin{aligned}
&\delta_t = r_{t+1} + \gamma Q^w(s_{t+1}, a_{t+1}) - Q^w(s_t, a_t),\\
&w_{t+1} = w_t + \alpha_w \delta_t \nabla_w Q^w(s_t, a_t),\\
&\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu_\theta(s_t) \nabla_a Q^w(s_t, a_t)|_{a=\mu_\theta(s_t)}.
\end{aligned} \tag{5.12}$$

Algorithm 5.1.2 (OPDAC(off-policy deterministic actor-critic))**.** *In this algorithm, we use Q-leanring to update critic, then*

$$\begin{aligned}
&\delta_t = r_{t+1} + \gamma Q^w(s_{t+1}, \mu_\theta(s_{t+1})) - Q^w(s_t, a_t),\\
&w_{t+1} = w_t + \alpha_w \delta_t \nabla_w Q^w(s_t, a_t),\\
&\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu_\theta(s_t) \nabla_a Q^w(s_t, a_t)|_{a=\mu_\theta(s_t)}.
\end{aligned} \tag{5.13}$$

Here, we should notice a problem that we can not use arbitrary approximator Q^w to approximate Q^μ, since $\nabla_a Q^w(s,a)$ may be different with $\nabla_a Q^\mu(s,a)$. The following theorem resolve this problem.

Theorem 5.1.3. *A function approximator Q^w is compatible with a deterministic policy μ_θ, $\nabla_\theta J(\theta) = \mathbb{E}_b[\nabla_\theta \mu_\theta(s) \nabla_a Q^w(s,a)|_{a=\mu_\theta(s)}]$,*

- $$\nabla_a Q^w(s,a)|_{a=\mu_\theta(s)} = \nabla_\theta \mu_\theta(s)^T w \tag{5.14}$$

- *w minimises the mean-squared error,* $\mathrm{MSE}(\theta, w) = \mathbb{E}[\varepsilon(s;\theta,w)^T \varepsilon(s;\theta,w)]$*, where* $\varepsilon(s;,\theta,w) = \nabla_a Q^w(s,a) - \nabla_a Q^\mu(s,a)$.

Proof: The proof is very simple. To minimize MSE, we wish

$$0 = \nabla_w \mathbb{E}[\varepsilon^T \varepsilon] = \mathbb{E}[\nabla_\theta \mu_\theta(s)^T w \varepsilon], \tag{5.15}$$

which means,

$$\mathbb{E}[\nabla_\theta \mu_\theta(s)^T \nabla_a Q^w(s,a)] = \mathbb{E}[\nabla_\theta \mu_\theta(s)^T \nabla_a Q^\mu(s,a)] = \nabla_\theta J(\theta). \tag{5.16}$$

QED.
And now, we should construct $Q^w(s,a)$ satisfying the first condition, the simple choice is,

$$Q^w(s,a) = (a - \mu_\theta(s))^T \nabla_\theta \mu_\theta(s)^T w + V^v(s), \tag{5.17}$$

where $V^v(s) = v^T \phi_s$ is a baseline only depending on s such that Q^w looks like an advantage function. For convenience, we denote $\phi(s,a) = \nabla_\theta \mu_\theta(s)(a - \mu_\theta)$. Hence,

$$Q^w = \phi(s,a)^T w + v^T \phi_s. \tag{5.18}$$

The second condition is very difficult to satifies in practice. Instead, we just use $Q-$learning, SARSA, or other algorithms to update w.

Algorithm 5.1.3 (COPDAC-Q)**.** *Compatible off policy deterministic actor-critic with Q-learning critic updates as,*

$$\delta_t = r_t + \gamma Q^w(s_{t+1}, \mu_\theta(s_{t+1})) - Q^w(s_t, \mu_\theta(s_t)), \tag{5.19}$$

$$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu_\theta(s)(\nabla_\theta \mu_\theta(s)^T w), \tag{5.20}$$

$$w_{t+1} = w_t + \alpha_w \delta_t \nabla_w Q^w = w_t + \alpha_w \delta_t \phi(s_t, a_t), \tag{5.21}$$

$$v_{t+1} = v_t + \alpha_v \delta_t \phi_{s_t} \tag{5.22}$$

It is well-known that off-policy Q-learning may diverage using linear function approximation. Hence, we use GTD to update critic by the following algorithm,

Algorithm 5.1.4 (COPDAC-GQ)**.** *Compatible off policy deterministic actor-critic with gradient Q-leanring algorithm(TDC) updates as,*

$$\delta_t = r_t + \gamma Q^w(s_{t+1}, \mu_\theta(s_{t+1})) - Q^w(s_t, \mu_\theta(s_t)), \tag{5.23}$$

$$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu_\theta(s)(\nabla_\theta \mu_\theta(s)^T w), \tag{5.24}$$

$$w_{t+1} = w_t + \alpha_w \delta_t \phi(s_t, a_t) - \alpha_w \gamma \phi(s_{t+1}, \mu_\theta(s_{t+1}))(\phi(s_t, a_t)^T u_t), \tag{5.25}$$

$$v_{t+1} = v_t + \alpha_v \delta_t \phi(s_t) - \alpha_v \gamma \phi(s_{t+1})(\phi(s_t, a_t)^T u_t), \tag{5.26}$$

$$u_{t+1} = u_t + \alpha_u(\delta_t - \phi(s_t, a_t)^T u_t)\phi(s_t, a_t). \tag{5.27}$$

DDPG algorithm

```
Randomly initialize critic network $Q^w(s,a)$ and $\pi_\theta(s)$ with weights $w$ and $\theta$.
Initialize target networl $Q_{\text{target}}$ and $\pi_{\text{target}}$ with $w^- \leftarrow w$ and $\theta^- \leftarrow \theta$;
Initialize replay buffer $B$
for episode $= 1, M$ do
    Initialise a noisy random process $\mathcal{N}$ for action exploration
    Receive initial observation states $s_1$
    for $t = 1, T$ do
        Select action $a_t = \pi(s_t, \theta) + \mathcal{N}$
        Store transition $(s_t, a_t, r_t, s_{t+1})$ in $B$.
        Sample random mini-batch of $N$ transitions $(s_i, a_i, r_i, s_{i+1})$ from $B$
        Set $y_i = r_i + \gamma Q_{\text{target}}(s_{i+1}, \pi_{\text{target}}(s_{i+1}))$.
        Update critic by minimising the loss $L(w) = \frac{1}{N}\sum_{i=1}^{N}(y_i - Q^w(s_i, a_i))^2$.
        Update actor policy using the sampled policy gradient:
            $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_q \nabla_a Q^(w)(s,a)|_{s=s_i,a=a_i} \nabla_\theta \pi_\theta(s)|_{s=s_i}$.
        Update target networks following soft updates:
            $w^{-1} \leftarrow \tau w + (1-\tau)w^-$
            $\theta^{-1} \leftarrow \tau\theta + (1-\tau)\theta^-$.
```

6 TRPO

TRPO(Trust Region Policy Optimization)[7] is a practical algorithm to update policy, which guarantees the monotonic improvement. The bais idea can be describe as: If we want to optimize an obejective function $f(x)$, which is very difficult to implement, we can improve the lower bound of this fucntion, for example, the expansion up to the second order for a local convex function as shown in Fig. The original objective author consider is expected discounted reward,

$$\eta(\pi) = \mathbb{E}_{s_0, a_0, \cdots \sim \pi}[\sum_{t=0}^{+\infty} \gamma^t r(s_t)], \tag{6.1}$$

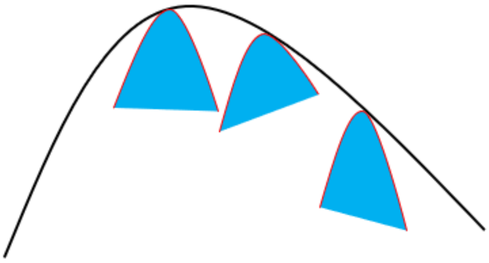

Figure 1. The black curse is our true objective, and we optimize red lower bound in local blue region.

where the initial state $s_0 \sim \rho_0(s)$. We can prove the difference between two policies $\pi, \widetilde{\pi}$ is,

$$\eta(\widetilde{\pi}) = \eta(\pi) + \mathbb{E}_{s_0,a_0,\cdots\sim\widetilde{\pi}}[\sum_{t=0}^{+\infty}\gamma^t A_\pi(s_t, a_t)]. \tag{6.2}$$

Intutively, it can be explained as: We have two kinds of trajectories generated by $\pi, \widetilde{\pi}$, the difference is the accumulated advantage of each element in trajectory generated by $\widetilde{\pi}$. Besides, we can show the simple proof. $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s) = \mathbb{E}_{s'\sim P(\cdot|s,a)}[r(s) + \gamma V_\pi(s') - V_\pi(s)]$. To simplify the notation, we use τ denote the trajectory $s_0, a_0, \cdots$.

$$\begin{aligned}\mathbb{E}_{\tau\sim\widetilde{\pi}}\sum_{t=0}^{+\infty}\gamma^t A_\pi(s_t,a_t) &= \mathbb{E}_{\tau\sim\widetilde{\pi}}\sum_{t=0}^{+\infty}\gamma^t(r(s_t) + \gamma V_\pi(s_{t+1}) - V_\pi(s_t)) \\ &= \eta(\widetilde{\pi}) - \mathbb{E}_{s_0}[V_\pi(s_0)] \\ &= \eta(\widetilde{\pi}) - \eta(\pi)\end{aligned} \tag{6.3}$$

Also, we can rewrite Eq.(6.2) as,

$$\eta(\widetilde{\pi}) = \eta(\pi) + \sum_s \rho_{\widetilde{\pi}}(s)\sum_a \widetilde{\pi}(s|a)A_\pi(s,a), \tag{6.4}$$

where $\rho_{\widetilde{\pi}}$ is discounted density of state under policy $\widetilde{\pi}$. In practice, the optimization of Eq.(6.4) has possibility making errors because of approximation error. Instead, we use another function,

$$L_\pi(\widetilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s)\sum_a \widetilde{\pi}(a|s)A_\pi(s,a). \tag{6.5}$$

The author proved the following two theorem, which introduce two lower bounds of $L_\pi(\widetilde{\pi})$, and then we can optimize these two lower bounds. Before statement, we give a measure of two distributions, $D_{TV}(p\|q) = \frac{1}{2}\sum_i |p_i - q_i|$. Hence, the difference between two policies can be described as,

$$D_{TV}^{\max} = \max_s D_{TV}(\pi(\cdot|s), \widetilde{\pi}(\cdot|s)). \tag{6.6}$$

Theorem 6.0.1. *Let $\alpha = D_{TV}^{\max}(\widetilde{\pi}, \pi)$. Then the following bound holds:*

$$\eta(\widetilde{\pi}) \geq L_{\pi}(\widetilde{\pi}) - \frac{4\varepsilon\gamma}{(1-\gamma)^2}\alpha^2, \varepsilon = \max_{s,a} |A_{\pi}(s,a)| \tag{6.7}$$

Besides,

$$\eta(\widetilde{\pi}) = L_{\pi}(\widetilde{\pi}) - \frac{4\varepsilon\gamma}{(1-\gamma)^2} D_{KL}^{\max}(\pi, \widetilde{\pi}), \tag{6.8}$$

where D_{KL} is KL divergence.

Hence, the optimization problem can be stated as,

$$\max_{\theta} L_{\theta_{old}} - CD_{KL}^{\max}(\theta, \theta_{old}) \tag{6.9}$$

But if we directly take $C = \frac{4\varepsilon\gamma}{(1-\gamma)^2}$ and do optimization, it can be shown that step size is too small. Hence, author modify this problem slightly: 1) give a constraint on KL divergence; 2) replace $D_{KL}^{\max}$ by an averaged KL divergence $\overline{D}_{KL}^{\rho} = \mathbb{E}_{s\sim\rho}[D_{KL}(\pi(\cdot|s), \widetilde{\pi}(\cdot|s))]$.

$$\begin{aligned} &\max_{\theta} L_{\theta_{old}}(\theta) \\ &\text{s.t.} \overline{D}_{KL}^{\theta_{old}}(\theta, \theta_{old}) \leq \delta. \end{aligned} \tag{6.10}$$

We can make it more practical by using importance sampling, with an off policy q,

$$\begin{aligned} &\max_{\theta} \mathbb{E}_{s\sim\rho_{old}, a\sim q}[\frac{\pi_\theta(a|s)}{q(a|s)} Q_{\theta_{old}}(s,a)] \\ &\text{s.t.} \mathbb{E}_{s\sim\rho_{old}}[D_{KL}(\pi_{old}(\cdot|s), \pi(\cdot|s))] \leq \delta. \end{aligned} \tag{6.11}$$

And now, we can discuss how to solve optimization problem Eq.(6.10), at least for a local region. Hence, we can expand the objective and constraint,

$$\begin{aligned} &\max_{\delta\theta} g^T \delta\theta \\ &\frac{1}{2}\delta\theta^T F \delta\theta \leq \delta. \end{aligned} \tag{6.12}$$

where $g = \nabla L(\theta)$, F is fisher information matrix of KL divergence, and we ignore the Hessian of L. we can reformulate this problem by adding a Lagrangian multipier as,

$$\min_{\delta\theta} \frac{1}{2}\delta\theta^T F \delta\theta - g^T \delta\theta, \tag{6.13}$$

which can solved by conjugate gradient algorithm because of F is semi-positive definite.

7 PPO

PPO[8] is the simplified version of TRPO, since the latter is very computionally complex. In Policy gradient methods, the most commonly used gradient estimator has the form:

$$\widehat{g} = \mathbb{E}_t[\nabla_\theta \pi_\theta(a_t|s_t)\widehat{A}_t] \tag{7.1}$$

Also, use automatic differentiation software, we can optimize the following loss function instead,

$$L^{\mathrm{PG}} = \mathbb{E}_t[\log \pi_\theta(a_t|s_t)A_t]. \tag{7.2}$$

In TRPO, an objective function is maximized subject to a constraint on the size of the policy update,

$$\begin{aligned} &\max_\theta \mathbb{E}_t[\frac{\pi_\theta(a_t|s_t)}{\pi_{\mathrm{old}}(a_t|s_t)}\widehat{A}_t] \\ &\text{s.t.}\mathbb{E}_t[\mathrm{KL}[\pi_{\mathrm{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]] \leq \delta. \end{aligned} \tag{7.3}$$

The idea in TRPO suggests using a penalty instead of a constraint, i.e solving the unconstrained optimization problem,

$$\max_\theta \mathbb{E}_t[\frac{\pi_\theta(a_t|s_t)}{\pi_{\mathrm{old}}(a_t|s_t)} - \beta \mathrm{KL}\pi_{\mathrm{old}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]] \tag{7.4}$$

7.1 clipped Surrogate Objective

In this paper, the author raises another new objective defined as,

$$L^{CLIP}(\theta) = \mathbb{E}_t[\min(r_t(\theta)\widehat{A}_t, \mathrm{clip}(r_t(\theta), 1+\varepsilon, 1-\varepsilon)\widehat{A}_t)], \tag{7.5}$$

where $r_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)}$, and ε is hyperparameter, say, $\varepsilon = 0.2$. And now, we can combine everything together, the final objective is,

$$L_t^{\mathrm{CLIP+VF+entropy}} = \mathbb{E}_t[L_t^{\mathrm{CLIP}} - c_1 L_t^{\mathrm{VF}}(\theta) + c_2 S[\pi_\theta](s_t)], \tag{7.6}$$

where c_1, c_2 are coefficients, and S denotes an entropy bonus, and L_t^{VF} is a squared-error loss $(V_\theta(s_t) - V_t^{\mathrm{target}})^2$.

8 A3C

Deep neural networks give sufficient representations to value function and policy, but which causes training process quite unstable. There are many approaches raised to solve this problem, and they share the same idea: the sequence of observed data and encounted by an online RL agents is non-stationary, and online RL updates are strongly correlated. By storing the agent in an experience replay memory, the data can be batched or randomly sampled from different time-steps. Aggregating over memory in this way reduces non-stationary and decorrelates updateds, but at the same time limits method to off-policy reinforment learning algorithms. In this paper, author provide a very different paradigm for deep reinforcement learning. Instead of experience replay, we asynchronously excute multiple agents in parallel, on multiple instances of environment. This parallelism also decorrelates the agents' data into a more stationary process. since at any given time-step the parallel agents will be experiencing a variety of different state.

A3C algorithm

```
Initialise $\theta, \theta_v, T = 0$.
Initialise thread step $t \leftarrow 1$
Repeat:
    Reset gradients: $d\theta \leftarrow 0, d\theta_v \leftarrow 0$.
    Synchronize thread-specificc parameters $\theta' = \theta, \theta'_v = \theta_v$
    $t_{\text{start}} = t$
    Get state $s_t$
    Repeat:
        Perform $a_t$ according to $\pi_\theta(a_t|s_t)$
        Receieve $r_t$ and new state $s_{t+1}$
        $t \leftarrow t+1, T \leftarrow T+1$
    Until terminal $s_t$ or $t - t_{\text{start}} == t_{\max}$

            $R = \begin{cases} 0, & \text{forterminalstate} \\ V(s_t, \theta'_v) & \text{otherwise} \end{cases} \qquad (8.1)$

    for $i \in \{1, 2, \cdots, t_{\text{start}}\}$ do
        $R \leftarrow r_i + \gamma R$
        Accumlate gradient wrt $\theta'$: $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i; \theta')(R - V(s_i; \theta'_v))$
        Accumlate gradient wrt $\theta'_v$: $d\theta_v \leftarrow d\theta_v + \nabla_{\theta'_v}(R - V(s_i; \theta'_v))^2$
    end for
    Perform asynchronous update $\theta$ using $d\theta$ and $\theta_v$ using $d\theta_v$.
Until $T > T_{\max}$.
```

References

[1] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[2] Richard S Sutton, Hamid R. Maei, and Csaba Szepesvári. A convergent o(n) temporal-difference algorithm for off-policy learning with linear function approximation. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1609–1616. Curran Associates, Inc., 2009.

[3] 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*, pages 993–1000. ACM, 2009.

[4] Hamid Reza Maei. Gradient temporal-difference learning algorithms. 2011.

[5] Thomas Degris, Martha White, and Richard S Sutton. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*, 2012.

[6] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

[7] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

A Lists of Proofs

A.1 Eligibility Traces

A.1.1 Equivalence between forward and backward

We prove equivalence for off-line update, which means V_t keeps invariant during one epoch. For this case, we can easily represent G_t^{t+n} by $\delta_{t+k}, k = 0, 1, 2 \cdots$.

$$G_t^{t+n} - V_t(S_t) = \sum_{k=0}^{n-1} \gamma^k \delta_{t+k} \tag{A.1}$$

And then,

$$\begin{aligned} G_t^{\lambda} - V_t(S_t) &= (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{t+n} \\ &= (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \sum_{k=0}^{n-1} \gamma^k \delta_{t+k} \\ &= \sum_{k=0}^{+\infty} (\lambda\gamma)^k \delta_{t+k}. \end{aligned} \tag{A.2}$$

QED.

A.2 TRPO

A.2.1 Proof Th.6.0.1

Defined average advantage over a different policy as,

$$\overline{A}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}A_\pi(s,a) \tag{A.3}$$

Definition A.2.1 (α coupled policy)**.** *$(\pi,\widetilde{\pi})$ is an α coupled policy pair if it defines a joint distribution $(a,\widetilde{a})|s$, such that $P(a \neq \widetilde{a}|s) \leq \alpha, \forall s$. π and $\widetilde{\pi}$ denote the marginal distributions of a and $\widetilde{a}$, respectively.*

Lemma A.2.1. *Given that $\pi, \widetilde{\pi}$ are α coupled policies, for all s,*

$$|\overline{A}(s)| \leq 2\alpha \max_{s,a} |A_\pi(s,a)|. \tag{A.4}$$

Proof:

$$\begin{aligned} |\overline{A}(s)| &= |\mathbb{E}_{\widetilde{a}\sim\widetilde{\pi}}A_\pi(s,\widetilde{a})| \\ &= |\mathbb{E}_{(\widetilde{a},a)\sim(\widetilde{\pi},\pi)}[A_\pi(s,\widetilde{(a)}) - A_\pi(s,a)]| \\ &\leq 2\max_{s,a}|A_\pi(s,a)|. \end{aligned} \tag{A.5}$$

Lemma A.2.2. *Let $(\pi,\widetilde{\pi})$ be an α coupled policy pair. Then,*

$$|\mathbb{E}_{s_t\sim\widetilde{\pi}}[\overline{A}(s_t)] - \mathbb{E}_{s_t\sim\pi}[\overline{A}(s_t)]| \leq 4\alpha(1-(1-\alpha)^t)\max_{s,a}|A_\pi(s,a)|. \tag{A.6}$$

Proof: The expectation difference is caused by each sampled state $s_i, i < t$, which is different between two trajectories generated by $\widetilde{\pi}$ and π. Hence, we can introduce random variable n_t denote the different state number up to time t. Then,

$$|\mathbb{E}_{s_t\sim\widetilde{\pi}}[\overline{A}(s_t)] - \mathbb{E}_{s_t\sim\pi}[\overline{A}(s_t)]| = P(n_t > 0)|\mathbb{E}_{s_t\sim\widetilde{\pi}|n_t>0}[\overline{A}(s_t)] - \mathbb{E}_{s_t\sim\pi|n_t>0}[\overline{A}(s_t)]|. \tag{A.7}$$

Since each step, the different probability is samller than α, then $P(n_t > 0) \leq 1-(1-\alpha)^t$. And the second part is not greater than $4\alpha\max_{s,a}|A_\pi(s,a)|$ from Lemma.(A.2.1). QED.
Proof of Theorem: Use $\varepsilon = \max_{s,a}|A_\pi(s,a)|$

$$\begin{aligned} \eta(\widetilde{\pi}) - L_\pi(\widetilde{\pi}) &= \sum_{t=0}^{+\infty}\gamma^t|\mathbb{E}_{\tau|\widetilde{\pi}}\overline{A}(s_t) - \mathbb{E}_{\tau|\pi}\overline{A}(s_t)| \\ &\leq \sum_{t=0}^{t=\infty}\gamma^t 4\alpha\varepsilon(1-(1-\alpha)^t) \\ &\leq \frac{4\alpha^2\gamma\varepsilon}{(1-\gamma)^2} \end{aligned} \tag{A.8}$$

Obivously, if we take $\alpha = D_{TV}^{\max}$, everything keeps invariant.