Reinforcement Learning- Assignment 2

YaChu Yang
11571276
ya-chu.yang@student.uva.nl
JiunHan Chen
11649712
chen.jiunhan@gmail.com

December 3, 2018

Abstract

1 Gradient Descent Methods

Question 1.1

Assume $\hat{v}_\pi(S_t)$ is the average of the Monte Carlo target. Then

$$\hat{v}_\pi(s) = \frac{1}{n}\sum_{i=1}^{n} G_{t,i,S_t=s} \tag{1}$$

$$\mathbb{E}(\hat{v}_\pi(s)) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}(G_{t,i}|S_t = s) \tag{2}$$

$$= \frac{1}{n}\sum_{i=1}^{n} v_\pi(s) = \frac{1}{n} n v_\pi(s) \tag{3}$$

$$= v_\pi(s) \tag{4}$$

Therefore, the average of G_t, Monte Carlo target, is an unbiased estimator of $v_\pi(s)$.

Question 1.2

The DP target depends on the current weight, w_t, which implies the estimator of the target value is biased and the value of gradient descent is not correct. Therefore, this method results in semi-gradient method due to including part of incorrect gradient value.

Question 1.3

The semi-gradient methods has two main advantages: one is that they learn faster than gradient methods; and the second one is that they can do learning without waiting for the end of an episode.

Therefore, in the Mountain Car problem, the agent can't learn until the car reaching the goal, which could waste a lot of time. Furthermore, because the Monte Carlo target, G_t, is calculated at the end of the episode, which means that all the reward of the action is incorporated in G_t, then if G_t is high, value of bad actions taken in the episode could be overestimated. This also leads to slow learning.

2 Basis functions

Question 2.1

Imagine a MDP with 3 states. The feature vectors can be represented as following:

$$x(s_1) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \qquad x(s_2) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad x(s_3) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{5}$$

$\hat{v}(s, \boldsymbol{\omega})$ can be represented as:

$$\hat{v}(s, \boldsymbol{\omega}) = \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3 \tag{6}$$

The tabular method can be represented by:

$$V(s_1) = a, \quad V(s_2) = b, \quad V(s_3) = c \tag{7}$$

Tabular method only replace ω as a, b, c, which are different state-values.

Question 2.2

The feature vector can incorporate product of x, y to enable the interaction between two features, like $\mathbf{x}(x, y) = (1, x, y, xy, x^2, y^2, xy^2, x^2y, x^2y^2)^T$

Question 2.3

Suppose a feature vector is composed by features made up by order-n polynomial-basis for k variables in state space. Therefore, one feature vector contains $(n+1)^k$ different features. Then the size of the polynomial feature vector depends exponentially on number of variables, which is k in the example above.

Question 2.4

If we use linear method to formulate the basis function, we can set $\mathbf{w}$, a linear function of the weight vector, differently according to the importance of the features.

For instance, in this case, we can set the weights of features with more important variables higher, and the other weights can be initialized as zero.

Question 2.5

Radial basis functions' feature can have different type of response on the distance between the state, s, and the center state, c_t, for example, Gaussian. If we choose step function as the distance metric in RBFs, then we can view RBFs as coarse coding, in which each feature is encoded to either 0 or 1 instead of the value in the interval [0, 1].

3 Geometry of linear value-function approximation

1.

$$\overline{\delta_\omega}(s_0) = (B^\pi v_\omega)(s_0) - v_\omega(s_0) \tag{8}$$
$$\overline{\delta_\omega}(s_1) = (B^\pi v_\omega)(s_1) - v_\omega(s_1) \tag{9}$$
$$\tag{10}$$

and,

$$(B_\pi v)(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v(s')] \tag{11}$$

Because we only have one possible action and one possible next state for each state, which leads to:

$$\overline{\delta_\omega}(s_0) = v_\omega(s_1) - v_\omega(s_0) = 1 \cdot 2 - 1 \cdot 1 = 1 \tag{12}$$
$$\overline{\delta_\omega}(s_1) = v_\omega(s_0) - v_\omega(s_1) = 1 \cdot 1 - 1 \cdot 2 = -1 \tag{13}$$
$$\tag{14}$$

2.

$$\|\overline{\delta_\omega}\|_\mu^2 = \sum_s \mu(s)\overline{\delta_\omega}(s)^2 \tag{15}$$
$$= \sum_s \frac{1}{2}\overline{\delta_\omega}(s)^2 \tag{16}$$
$$= 1 \tag{17}$$

3.

$$\omega = \arg\min_\omega \|\overline{\delta_\omega}\|_\mu^2 \tag{18}$$
$$= \arg\min_\omega \sum_s \frac{1}{2}\overline{\delta_\omega}(s)^2 \tag{19}$$
$$= \arg\min_\omega \frac{1}{2}[(\omega\phi(s_1) - \omega\phi(s_0))^2 + (\omega\phi(s_0) - \omega\phi(s_1))^2] \tag{20}$$
$$\Rightarrow \omega = 0 \tag{21}$$

4.

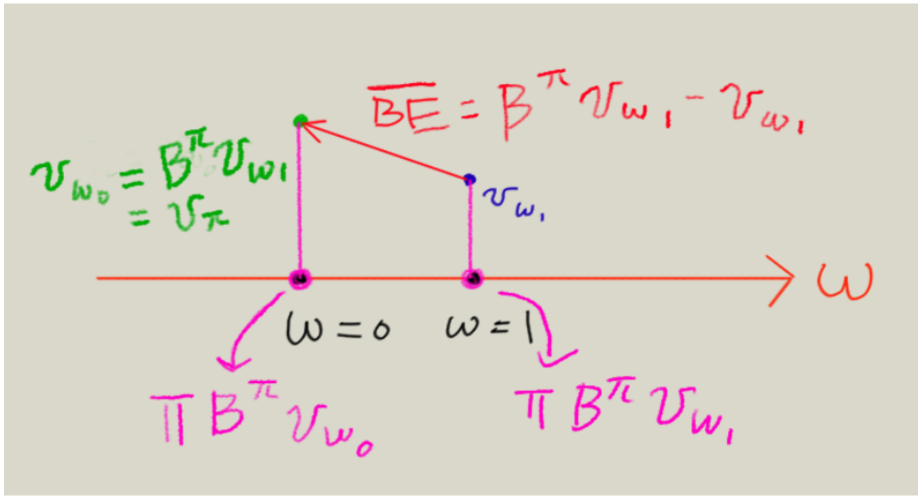


Figure 1: The geometry of linear value-function approximation

Figure 1 shows after initialization $\omega = 1$, the value-function has difference from v_π. We apply bellman operator to $v_{\omega=1}$, the value-function becomes $v_{\omega=0}$. For $v_{\omega=0}$, it corresponds to $\omega = 0$ after projected to parameter space, which coincidentally equals to v_π, because the reward is always zero in this case and $\omega = 0$ make value-function zero too. So after applying bellman equation(bell operation), our original v_ω indeed becomes closer to v_π.

4 Neural Networks

1.
The state distribution $\mu(s)$ depends on the frequency we visit the state s. When we update parameters of the value function approximator, the output of value function approximator will change. For DQN, value function approximator can decide which actions will be selected, so updating parameters leads to the change of state distribution too.

2.
For standard (un-)supervised learning problems, the inputs doesn't change when the model parameters change. But for the value function approximator, the change of parameters leads to the inputs(the states) change, which might lead to the difficulty of training our model.

3.
It means that the error might be changed not by minimizing the error of each state but by adjusting state distribution.

4.
Sequential data like $(s_0, a_0, R_1, s_1, a_1 ...)$ has high correlation. For neural network, if our train data has high correlation, it will cause gradient focus on limited update direction. Experience replay can save the progress of playing into memory, and we can extract it from the memory when we

want to train our model by the experience before, which reduce the dependency of our sequential data.

5.
This trick is also for reducing the correlation[1] as citation mentioned in page 1. It uses two network, one is evaluation network, another is target network. The target network is only periodically updated by copying the parameters from evaluation network, which can make the update of $Q(s,a) = r + \gamma max_{a'} Q(s', a')$ reduce correlation of adjacent states.

5 Policy Gradient

Question 5.1 REINFORCE

1.

$$VAR\big((G(\tau) - b)\nabla logp(\tau)\big) = \mathbb{E}_\tau\Big[\big((G(\tau) - b)\nabla logp(\tau)\big)^2\Big] - \mathbb{E}_\tau\big[(G(\tau) - b)\nabla logp(\tau)\big]^2$$
$$= \mathbb{E}_\tau\big[G^2(\tau)(\nabla logp(\tau))^2\big] + b^2\mathbb{E}[(\nabla logp(\tau))^2] - 2b\mathbb{E}[G(\tau)(\nabla logp(\tau))^2] - \mathbb{E}_\tau\big[(G(\tau) - b)\nabla logp(\tau)\big]^2$$

b* which minimizes the variance satisfies:

$$0 = \frac{dVAR\big((G(\tau) - b)\nabla logp(\tau)\big)}{db} = 2b\mathbb{E}[(\nabla logp(\tau))^2] - 2\mathbb{E}[G(\tau)(\nabla logp(\tau))^2]$$
$$\therefore b = \frac{\mathbb{E}[G(\tau)(\nabla logp(\tau))^2]}{\mathbb{E}[(\nabla logp(\tau))^2]}$$

2.

$$\begin{aligned} b &= \frac{\mathbb{E}[G(\tau)(\nabla logp(\tau))^2]}{\mathbb{E}[(\nabla logp(\tau))^2]} = \frac{\mathbb{E}[(a+2)(a-\theta)^2]}{\mathbb{E}[(a-\theta)^2]} \\ &= \frac{\mathbb{E}(a^3 + a\theta^2 - 2a^2\theta)}{\mathbb{E}[a^2 + \theta^2 - 2a\theta]} + 2 \\ &= \frac{\theta + \theta^3 - 2\theta^2}{\theta + \theta^2 - 2\theta^2} + 2 = \frac{\theta(1-\theta)^2}{\theta(1-\theta)} + 2 = 3 - \theta \end{aligned}$$

3.

$$\mathbb{E}_\tau[\sum_{t=1}^{T} \nabla \log \pi(a_t|s_t) b(s_t)] = \mathbb{E}_\tau[\nabla \log \pi(a_1|s_1)b(s_1)] + \mathbb{E}_\tau[\nabla \log \pi(a_2|s_2)b(s_2)] + ... + \mathbb{E}_\tau[\nabla \log \pi(a_T|s_T)b(s_T)]$$

We compute $\mathbb{E}_\tau[\nabla \log \pi(a_k|s_k)b(s_k)] \quad k = 1, 2, ..., T$:

$$\begin{aligned}
\mathbb{E}_\tau[\nabla \log \pi(a_k|s_k)b(s_k)] &= \int p(\tau)\nabla \log \pi(a_k|s_k)b(s_k)d\tau \\
&= \int \prod_{t=1}^{T} \mu(s_t)\pi(a_t|s_t)\nabla \log \pi(a_k|s_k)b(s_k)d\tau \\
&= \int\int\cdots\int b(s_k)\mu(s_k)[\int \pi(a_k|s_k)\nabla \log \pi(a_k|s_k)da_k]ds_k d\tau_{-k} \\
&= \int\int\cdots\int b(s_k)\mu(s_k)\underbrace{[\nabla \int \pi(a_k|s_k)da_k]}_{0} ds_k d\tau_{-k} \\
&= 0
\end{aligned}$$

So,

$$\mathbb{E}_\tau[\sum_{t=1}^{T} \nabla \log \pi(a_t|s_t)b(s_t)] = 0$$

Question 5.2 Compatible Function Approximation Theorem

1.

$$\begin{aligned}
\textit{Given a s},\ \mathbb{E}_a[\hat{q}_w(s,a)] &= \mathbb{E}_a[w^T \nabla_\theta log_\pi(s,a)] \\
&= \int \pi(a|s)w^T \nabla_\theta log_\pi(a|s)da \\
&= w^T \int \pi(a|s)\frac{\nabla_\theta \pi(a|s)}{\pi(a|s)}da \\
&= w^T \nabla_\theta \int \pi(a|s)da \\
&= w^T \nabla_\theta 1 = 0 \\
\therefore \mathbb{E}_a[\hat{q}_w(s,a)] &= 0, \forall s \in S.
\end{aligned}$$

All expected value of $q_w(s,a)$ equal to zero can be implicitly interpreted as that we subtract baselines according to each states to every q. By doing subtracting a baseline, q_w can be viewed as the relative value, instead of the absolute value.

2.

$$\begin{aligned}
\mathbb{E}_a[q_\pi(s,a) - v_\pi(s)] &= \int \pi(a|s)(q_\pi(s,a) - v_\pi(s))da \\
&= \int \pi(a|s)q_\pi(s,a)da - v(s)\int \pi(a|s)da \\
&= v(s) - v(s) = 0
\end{aligned}$$

3. Combining $\mathbb{E}_a[\hat{q}_w(s,a)] = 0$ and $\mathbb{E}_a[q_\pi(s,a) - v_\pi(s)] = 0$, we can consider $\hat{q}_w(s,a)$ as an approximation of advantage function rather than $q_\pi(s,a)$. $\mathbb{E}_a[q_\pi(s,a)]$ might be not zero.

4.
We must create $\hat{q}_w(s,a)$, which satisfies $\nabla_\omega \hat{q}_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$

$$\begin{aligned}
\nabla_\theta \log \pi_\theta(s,a) &= \nabla_\theta(\theta^T \phi_{sa} - \log \sum_b e^{\theta^T \phi_{sb}}) \\
&= \phi_{sa} - \sum_b \frac{e^{\theta^T}}{\sum_c e^{\theta^T \phi_{sc}}} \phi_{sb} \\
&= \phi_{sa} - \sum_b \pi_\theta(s,b)\phi_{sb}
\end{aligned}$$

So we can create:

$$\hat{q}_w(s,a) = \omega^T(\phi_{sa} - \sum_b \pi_\theta(s,b)\phi_{sb})$$

, which satisfies $\nabla_\omega \hat{q}_w(s,a) = \nabla_\theta \log \pi_\theta(s,a)$

Question 5.3 Natural Gradient

1.

$$\begin{aligned}
log\pi(a|\theta) &= -log(exp(\theta_\sigma)) - log(\sqrt{2\pi}) - \frac{1}{2}(\frac{(a-\theta_\mu)}{(exp(\theta_\sigma))})^2 \\
&= -\theta_\sigma - log(\sqrt{2\pi}) - \frac{1}{2}(\frac{(a-\theta_\mu)}{(exp(\theta_\sigma))})^2 \\
\nabla_{\theta\mu} log\pi(a|\theta) &= \frac{-2(a-\theta_\mu)}{2[exp(\theta_\sigma)]^2} = \frac{a-\theta_\mu}{[exp(\theta_\sigma)]^2} \\
\nabla_{\theta\sigma} log\pi(a|\theta) &= -1 - [-2\frac{(a-\theta_\mu)^2}{2}(exp(\theta_\sigma))^{-3} exp(\theta_\sigma)] \\
&= -1 + \frac{(a-\theta_\mu)^2}{(exp(\theta_\sigma))^2}
\end{aligned}$$

2. In vanilla gradient descent, the parameters are updated in the direction of the gradient by a small distance, which is defined by Euclidean distance in the parameter space. However, not all parameters should be updated equally.Therefore, rather than using vanilla gradient descent, using

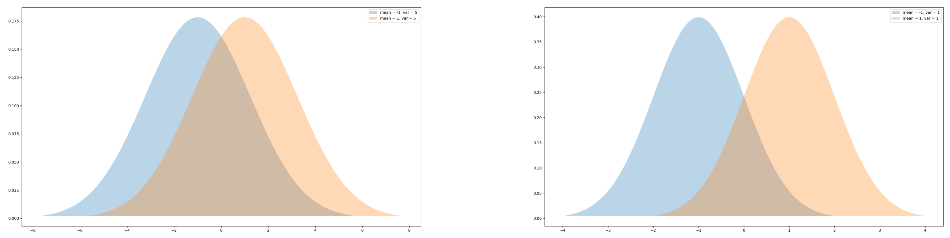
natural gradient, we can treat the change of every parameters separately to make the network's entire output distribution stay similar during the learning of the parameters.

3.

$$\begin{aligned}
g(\theta, a) &= \nabla_{d\theta} log\pi(a|\theta_0 + d\theta)\nabla_{d\theta} log\pi(a|\theta_0 + d\theta)^T \\
&= \begin{bmatrix} \frac{a-\theta_\mu}{[exp(\theta_\sigma)]^2} \\ -1 + \frac{(a-\theta_\mu)^2}{(exp(\theta_\sigma))^2} \end{bmatrix} \begin{bmatrix} \frac{a-\theta_\mu}{[exp(\theta_\sigma)]^2} & -1 + \frac{(a-\theta_\mu)^2}{(exp(\theta_\sigma))^2} \end{bmatrix} \\
&= \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \\
\mathbb{E}[g_{11}] &= \mathbb{E}\big(\frac{(a-\theta_\mu)^2}{(exp(\theta_\sigma))^4}\big) = \frac{1}{(exp(\theta_\sigma))^4}\big(\mathbb{E}[a^2] - 2\mathbb{E}[a]\theta_\mu + \theta_\mu^2\big) \\
&= \frac{1}{(exp(\theta_\sigma))^4}\big(\theta_\mu^2 + (exp(\theta_\sigma))^2 - 2\theta_\mu^2 + \theta_\mu^2\big) \\
&= \frac{1}{(exp(\theta_\sigma))^2} \\
\mathbb{E}[g_{12}] = \mathbb{E}[g_{21}] &= \mathbb{E}[-\frac{1}{(exp(\theta_\sigma))^2}(a-\theta_\mu) + \frac{1}{(exp(\theta_\sigma))^3}(a-\theta_\mu)^3] \\
&= -\frac{1}{(exp(\theta_\sigma))^2}\mathbb{E}[a-\theta_\mu] + \frac{1}{(exp(\theta_\sigma))^3}\mathbb{E}[(a-\theta_\mu)^3] \\
&= \frac{1}{(exp(\theta_\sigma))^3}\mathbb{E}[(a-\theta_\mu)^3] = \frac{1}{(exp(\theta_\sigma))^3}\mathbb{E}[a^3 + 3a\theta_\mu^2 - 3a^2\theta_\mu - \theta_\mu^3] \\
&= \frac{1}{(exp(\theta_\sigma))^3}\big(\mathbb{E}[a^3] + 3\mathbb{E}[a]\theta_\mu^2 - 3\mathbb{E}[a^2]\theta_\mu - \theta_\mu^3\big) \\
&= \frac{1}{(exp(\theta_\sigma))^3}\big(\theta_\mu^3 + 3\theta_\mu(exp(\theta_\sigma))^2 + 3\theta_\mu\theta_\mu^2 - 3(\theta_\mu^2 + (exp(\theta_\sigma))^2)\theta_\mu - \theta_\mu^3\big) = 0 \\
\mathbb{E}[g_{22}] &= \mathbb{E}[(-1 + \frac{(a-\theta_\mu)^2}{(exp(\theta_\sigma))^2})^2] = \mathbb{E}[1 - 2\frac{(a-\theta_\mu)^2}{(exp(\theta_\sigma))^2} + (\frac{(a-\theta_\mu)^4}{(exp(\theta_\sigma))^4})] \\
&= 1 - \frac{2}{(exp(\theta_\sigma))^2}(exp(\theta_\sigma))^2 + \frac{1}{(exp(\theta_\sigma))^4}[\mathbb{E}[a^4 - 4a^3\theta_\mu + 6\theta_\mu^2 a^2 - 4\theta_\mu^3 a + \theta_\mu^4]] \\
&= 1 - 2 + \frac{1}{(exp(\theta_\sigma))^4}\big[\mathbb{E}[a^4] - 4(\theta_\mu^3 + 3\theta_\mu\theta_\sigma^2)\theta_\mu + 6\theta_\mu^2(\theta_\mu^2 + \theta_\sigma^2) - 4\theta_\mu^4 + \theta_\mu^4\big] \\
&= -1 + \frac{1}{(exp(\theta_\sigma))^4}\big[\mathbb{E}[a^4] - \theta_\mu^4 - 6\theta_\mu^2\theta_\theta^2\big] \\
&= -1 + \frac{1}{(exp(\theta_\sigma))^4}\big[3(exp(\theta_\sigma))^4 + \theta_\mu^4 + 6\theta_\mu^2\theta_\theta^2 - \theta_\mu^4 - 6\theta_\mu^2\theta_\theta^2\big] = 2 \\
\therefore F &= \begin{bmatrix} \frac{1}{(exp(\theta_\sigma))^2} & 0 \\ 0 & 2 \end{bmatrix}
\end{aligned}$$

4. From $\theta^* - \theta_0 \propto F^{-1}\nabla_\theta J(\theta_0)$, we can know that the update step for θ_μ depends on the standard deviation of the policy, $\sigma(\theta_\sigma)$. When $\sigma(\theta_\sigma)$ is small, the update step for θ_μ will be small, and vice

versa. This can be explained in Figure 2. In Figure 2a and Figure 2b, though their change of mean is from -1 to 1, we can see that the first distribution changed far less than the second. Therefore, to keep the old and new distributions similar to each other, when the standard deviation of the policy is small, the update step should be also small too, and the other way around.



(a) Gaussian distributions with large standard deviation.
(b) Gaussian distributions with small standard deviation.

Figure 2: Illustration of same step size ending up with different change of the distributions.

5.
$\nabla_{\theta_\sigma} log\pi(a|\theta) = \frac{-1}{\theta_\sigma} + \frac{(a-\theta_\mu)^2}{\theta_\sigma^3}$
$\nabla_{\theta\mu} log\pi(a|\theta) = \frac{a-\theta_\mu}{[(\theta_\sigma)]^2}$

The standard deviation and the Fisher information matrix calculated in subsection 5.3.3 are denoted as σ'_θ, F' .

$$\text{According to } F'_{11} = \mathbb{E}[g_{11}]\text{, calculated in subsection 5.3.3,} \quad F_{11} = \frac{1}{\theta_\sigma^2}$$
$$F_{12} = F_{21} = 0$$

According to subsection 5.3.3,

$$\nabla_{\theta'_\sigma} log\pi(a|\theta') = -1 + \frac{(a-\theta_\mu)^2}{{\sigma'_\theta}^2}$$
$$\nabla_{\theta_\sigma} log\pi(a|\theta) = \frac{-1}{\sigma_\theta} + \frac{(a-\theta_\mu)^2}{\sigma_\theta^3}$$

$\therefore$ We just need to multiple $\frac{1}{\sigma_\theta^2}$ with F'_{22} and replace σ'_θ to σ_θ we can get new F_{22}

$$\therefore F_{22} = \frac{2}{\theta_\sigma^2}$$
$$F = \begin{bmatrix} \frac{1}{\theta_\sigma^2} & 0 \\ 0 & \frac{2}{\theta_\sigma^2} \end{bmatrix}$$

6. When the parameterizations of σ change, the right corner element in Fisher information matrix's dependence on σ changes. In previous example, the element changes from 2 to $\frac{2}{\sigma^2}$

7. For the exponential sigma with Fisher matrix:

$$\begin{aligned}
\sigma = 4 &= e^{\theta_\sigma} \\
\Rightarrow \theta_\sigma &= \log 4 \\
\theta_\mu &= \mu = 0
\end{aligned}$$

$$F = \begin{bmatrix} \frac{1}{(exp(\theta_\sigma))^2} & 0 \\ 0 & 2 \end{bmatrix} \Rightarrow F^{-1} = \begin{bmatrix} 16 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

$$\begin{aligned}
\nabla_{\theta_\sigma} J &= G(\tau) \nabla_{\theta_\sigma} \log \pi(a = 8 | \mu = 0, \sigma = 4) \\
&= \log \pi(a = 8 | \theta_\mu = 0, \theta_\sigma = \log 4) \\
&= -1 + \frac{(a - \theta_\mu)^2}{(exp(\theta_\sigma))^2} \\
&= 3
\end{aligned}$$

$$\begin{aligned}
\theta_{new} &= \theta + \alpha F^{-1} \nabla_\theta J(\theta_0) \\
\theta_{\sigma new} &= \log 4 + 0.01 * \frac{1}{2} * 3 \\
&= 1.401
\end{aligned}$$

For the exponential sigma with vanilla policy gradient:

$$\begin{aligned}
\theta_{\sigma new} &= \log 4 + 0.01 * 3 \\
&= 1.416
\end{aligned}$$

For linear sigma with Fisher matrix:

$$F = \begin{bmatrix} \frac{1}{\theta_\sigma^2} & 0 \\ 0 & \frac{2}{\theta_\sigma^2} \end{bmatrix} \Rightarrow F^{-1} = \begin{bmatrix} 16 & 0 \\ 0 & 8 \end{bmatrix}$$

$$\begin{aligned}
\nabla_{\theta_\sigma} J &= \nabla_{\theta_\sigma} log \pi(a | \theta) \\
&= \frac{-1}{\theta_\sigma} + \frac{(a - \theta_\mu)^2}{\theta_\sigma^3} \\
&= \frac{3}{4}
\end{aligned}$$

$$\begin{aligned}\theta_{\sigma new} &= 4 + 0.01 * 8 * 3/4 \\ &= 4.06\end{aligned}$$

For linear sigma without Fisher matrix:

$$\begin{aligned}\theta_{\sigma new} &= 4 + 0.01 * 3/4 \\ &= 4.0075\end{aligned}$$

References

[1] 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, 2015.