Reinforcement Learning - Assignment 2
Group 3

Davide Belli
11887532
University of Amsterdam
davide.belli@student.uva.nl

Gabriele Cesa
11887524
University of Amsterdam
gabriele.cesa@student.uva.nl

2 Gradient Descent Methods

2.1

Real target is obtained through real rewards received by interacting with the environment in the episode (unbiased), and not estimates of the rewards from the value function learned so far (biased).

2.2

Since the targets are biased due to using the bootstrapping for the value function parametrized through $\mathbf{w}_t$, both targets and estimates will depend on the function approximating the value function. As a result, we use the semi-gradient method to only compute gradient wrt the estimated reward.

2.3

One of the problems that Monte Carlo approaches face in the Mountain Car problem is that the reward and the termination of the episode are only seen when the car reaches the top of the hill. This, of course, is going to happen very rarely, due to the complexity of the sequence of actions leading to the car reaching the top of the mountain (even with a very explorative policy). Although bootstrapping-based methods are biased, using the estimation of future rewards instead of the actual rollout of the true episode allows for a much faster learning in this task. The policy is updated at every time-step in the simulation by considering possible bootstrapped future states. Using the semi-gradient instead of the full gradient, the learning of the optimal policy will be even faster (less computations for the backpropagation steps).

3 Basis functions

3.1

When using tabular methods, we explicitly build a dictionary mapping a discrete number of states to their values. This is completely equivalent to use a one-hot encoding of the states in a linear method, where we associate to each dimension of the feature vector a different state. A feature vector will contain a 1 in the position corresponding to the current state and 0 elsewhere. As a result, a linear combination of the features will result in just selecting one of the weights in $\boldsymbol{w}$ (all the others will be multiplied by 0). This particular weight will be equal to the linear combination and, so, it will correspond to the estimated value for the current state. Indeed, notice that the vector $\boldsymbol{w}$ will store one weight for each possible state and that each weight-value is used only when we are in its corresponding state.

3.2

Given a state space defined by two features, $s = [x, y]$, we can design polynomial features for this state space in order to model their interaction. To do this, we can define basis functions as the set or subset of all the possible combinations of the polynomials for the two feature up to some degree n. For example, considering all the polynomials up to degree $n = 2$, the feature vector including the set of combinations will result in $s' = [1, x, y, xy, x^2, y^2]$. In general, if we need powers up to n, we can pick monomials from the set $S_n = \{x^a y^b : a + b \leq n \wedge a, b \geq 0\}$. If we know something about the dimensions of the problem, we can pick the most meaningful combinations of features, or at least define up to which polynomial degree we are interested.

3.3

If the number of variables in our state space increases linearly, the size of the polynomial feature vector including all the combinations up to some degree n will increase exponentially.

3.4

There are different ways to encode the prior knowledge about what dimension is more relevant when defining the basis functions. If we are considering polynomial basis function, we may want to have most of the functions to include this feature with some exponent. (E.g., if we use the set $s = [a, b, c, d, e]$ and know that a is very relevant to our task, we may want to use the basis functions $s' = [1, a, a^2, ab, ac, ad, ae, ...]$ rather than focusing on the other features). When using the Fourier Basis, we may want to define a large range of different frequency coefficients for the most important dimension, so that we may be able to have an accurate discrimination of this dimension both at high granularity (smaller changes, high frequencies) and at coarser level (larger changes, low frequencies) for the values assumed by that feature. The concept of emphasizing the importance of a dimension when building the basis function can be extended also to other type of basis, like Tiling and RBFs. For instance, we could use RBFs with deformed receptive fields such that they have larger variance in the direction of the less important variable and we could place more RBFs along the important dimension to give a finer representation of the values it can take.

3.5

Coarse coding can be seen as a special case of RBFs. In Coarse coding, instead of having smooth values in the interval $[0, 1]$ to describe the degree to which a feature is present (for example by using a Gaussian response for the original features), the presence of a feature is modeled in a binary way, either present or absent (numerically, the feature for the current datapoint assumes values $\{0, 1\}$ depending on the input values from the state space). This can also be interpreted as associating to each RBF a threshold and assigning a point to all RBFs (setting the corresponding feature to 1) whose values, evaluated on that point, are above those thresholds.

4 Geometry of linear value-function approximation

4.1

By initializing the weight $w = 1$ and using $\gamma = 1$ we can compute the initial approximation of state values as:

$$v_w(\mathbf{s}) = w \cdot \boldsymbol{\phi} = 1 * \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \tag{1}$$

Then, we can compute the Bellman error as:

$$\overline{\delta}_w = B^\pi v_w - v_w = \begin{bmatrix} r_1 + \gamma * v_w(s_2) - v_w(s_1) \\ r_2 + \gamma * v_w(s_1) - v_w(s_2) \end{bmatrix} = \begin{bmatrix} 0 + 1 * 2 - 1 \\ 0 + 1 * 1 - 2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{2}$$

4.2

$$\overline{BE}(w) = ||\bar{\delta}_w||^2 = \frac{1}{2}1^2 + \frac{1}{2}(-1)^2 = 1 \tag{3}$$

4.3

First, let's do a step with the Bellman's operator:

$$B^\pi v_w = \begin{bmatrix} r_1 + \gamma * v_w(s_2) \\ r_2 + \gamma * v_w(s_1) \end{bmatrix} = \begin{bmatrix} 0 + 1 * 2 \\ 0 + 1 * 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \tag{4}$$

Now we want to find the w^* such that the new v_{w^*} is as close as possible to the $B^\pi v_w$ just computed, i.e.:

$$v_{w*} = \Pi B^\pi v_w \tag{5}$$

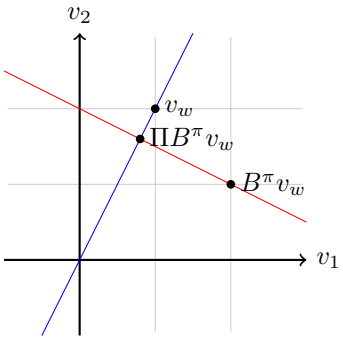
where

$$\begin{aligned} w^* &= argmin_{w^*} ||B^\pi v_w - v_{w*}||^2 \\ 0 &= \frac{\partial ||B^\pi v_w - v_{w*}||^2}{\partial w^*} \\ 0 &= \frac{\partial \left((2 - w^*)^2 + (1 - 2w^*)^2\right)}{\partial w^*} \\ 0 &= 10w^* - 8 \\ w^* &= \frac{4}{5} \end{aligned}$$

then

$$\Pi B^\pi v_w = v_{w^*} = \begin{bmatrix} \frac{4}{5} \\ \frac{8}{5} \end{bmatrix} \tag{6}$$

4.4



Considering the plot above, we first show the value of the state with the initial state values v_w for the parameter $w = 1$. Then, we plot the target value $B^\pi v_w$ applying Bellman's operator. With a blue line we show the 1-dimensional manifold of state values that can be obtained changing the w parameter for our function approximation. By applying the projection operator on the Bellman's target value, we find the parameter value $w^* = \frac{4}{5}$ which minimizes the distance between the target value and the estimate value. This behavior can be confirmed graphically visualizing how the projected point corresponds to the geometrical projection of $B^\pi v_w$ on the blue line representing our function approximator manifold. Notice how the point-to-line distance segment lays on the red line, which is perpendicular to the blue manifold line.

5 Neural Networks

5.1

When we train a function approximator with a learnable set of parameters, we try to minimize a certain error. For example, commonly used error metrics are the Mean Squared Value Error $\overline{VE}$ and the Mean Squared TD Error $\overline{TDE}$. In both cases the state distribution is used to weight the value estimate error (which depends on the weights of the function approximator). Indeed, we prefer having a more accurate approximation of frequently occurring states rather than rarely occurring ones. By updating the weights of the learnable function approximator in order to minimize the error we also change the learned policy (in on-line learning), which in turn, results in a modification in the state distribution. At the end of the training both the weight values and the state distribution will converge, meaning that the state distribution $\mu(s)$ will be a stationary distribution resulting from the policy depending on final values of the weights $\mathbf{w}$ learned during the process.

5.2

The main difference with standard (un-)supervised learning is that the latter assumes a static training dataset over which multiple passes can be made, while in reinforcement learning the training happens online. Moreover, a common assumption in supervised learning is the independence of the data in the dataset, whereas in a common reinforcement learning task we have a sequence of observations which depend on each other within a single episode, but we may still want to learn during the episode. In the end, as discussed in the previous answer, the state distribution is learned online and depends on the learned model (function approximator and policy) itself, while in (un-)supervised learning the optimal model depend on a static dataset of which we already know the class distribution

Another important difference is that in reinforcement learning, it is not necessary true that the model which best approximates the value function is the most preferable solution. Indeed, the final goal is to build the best policy, not the best value estimation.

5.3

As a result, approximation errors made in states which are very little likely to be visited are less important than errors made in frequently visited states. Indeed, eq. 9.1 weights the errors in each state by the estimated probability of observing that state. In comparison, weighting in (un-)supervised learning can be done according to classes' distributions to balance the frequency of rare classes or, on the other way, to improve accuracy in some classes when they are more important than others. The knowledge of the classes distribution based on which the weighting is done is known a priori, and does not depend on the learned model. As we discussed in the previous answers, this is not the case for reinforcement learning where the final stationary state distribution depends on the learned function approximator (and extracted policy).

5.4

One of the strongest assumption used while training deep network (and many of the most popular supervised learning algorithms) is the IID of the datapoints in the dataset (datapoints are generated according to the same distribution independently from each other). However, in a reinforcement learning task we observe several episodes, each of which is made of a sequence of (state, action, reward, next state) tuples. Though different episodes can be considered independent from each other, the steps within an episode are strongly correlated (assuming a Markov Process, each step depends on the previous one). As a result, using batches of consecutive steps of an episode breaks the IID assumption necessary for training these kinds of models.

A solution commonly employed is the *experience replay*: while simulating an episode, at each step, instead of training on the current observation we store it in a separate memory. Then, in order to train a model, we sample a mini-batch from this memory and feed it to the model. The training step can be done after each step in the simulation or at different frequencies. In this way, the mini-batch can contain independent samples from the memory.

5.5

In the DQN paper, Mnih et al. also suggest freezing the target network for periods of time. The intuition behind this technique is that we want to break the correlation between the Q-network estimate and targets. At the same time, since the target network is only updated to the current network state at fixed training steps, the training will be more stable because oscillations in updates are prevented.

6 Policy Gradient

6.1 REINFORCE

6.1.1

We assume there is only 1 parameter.

$$
\begin{aligned}
& Var_\tau\left((G(\tau)-b)\nabla \log p(\tau)\right) \\
=& \mathop{\mathbb{E}}_{\tau}[((G(\tau)-b)\nabla \log p(\tau))^2] - \mathop{\mathbb{E}}_{\tau}[(G(\tau)-b)\nabla \log p(\tau)]^2 \\
=& \mathop{\mathbb{E}}_{\tau}[(G(\tau)-b)^2\nabla \log p(\tau)^2] - (\nabla J)^2 \\
=& \mathop{\mathbb{E}}_{\tau}[(G(\tau)^2 - 2bG(\tau) + b^2)\nabla \log p(\tau)^2] - (\nabla J)^2 \\
=& \mathop{\mathbb{E}}_{\tau}[G(\tau)^2\nabla \log p(\tau)^2] - 2b\mathop{\mathbb{E}}_{\tau}[G(\tau)\nabla \log p(\tau)^2] \\
& + b^2\mathop{\mathbb{E}}_{\tau}[\nabla \log p(\tau)^2] - (\nabla J)^2 \\
=& \mathop{\mathbb{E}}_{\tau}[G(\tau)^2\nabla \log p(\tau)^2] - (\nabla J)^2 \\
& - 2b\mathop{\mathbb{E}}_{\tau}[G(\tau)\nabla \log p(\tau)^2] + b^2\mathop{\mathbb{E}}_{\tau}[\nabla \log p(\tau)^2] \\
=& Var_\tau(G(\tau)\nabla \log p(\tau)) - 2b\mathop{\mathbb{E}}_{\tau}[G(\tau)\nabla \log p(\tau)^2] + b^2\mathop{\mathbb{E}}_{\tau}[\nabla \log p(\tau)^2]
\end{aligned}
$$

In order to minimize the variance:

$$
\begin{aligned}
\frac{\partial Var_\tau\left((G(\tau)-b)\nabla \log p(\tau)\right)}{\partial b} &= 0 \\
-2\mathop{\mathbb{E}}_{\tau}[G(\tau)\nabla \log p(\tau)^2] + 2b\mathop{\mathbb{E}}_{\tau}[\nabla \log p(\tau)^2] &= 0 \\
\mathop{\mathbb{E}}_{\tau}[G(\tau)\nabla \log p(\tau)^2] &= b\mathop{\mathbb{E}}_{\tau}[\nabla \log p(\tau)^2] \\
b &= \frac{\mathbb{E}_\tau[G(\tau)\nabla \log p(\tau)^2]}{\mathbb{E}_\tau[\nabla \log p(\tau)^2]}
\end{aligned}
$$

6.1.2

A trajectory τ is composed of a single action a. Therefore: $G(\tau) = a + 2$ and $p(\tau) = p(a)$. Then:

$$
\begin{aligned}
b &= \frac{\mathbb{E}_\tau[G(\tau)\nabla \log p(\tau)^2]}{\mathbb{E}_\tau[\nabla \log p(\tau)^2]} \\
&= \frac{\mathbb{E}_a[G(a)\nabla \log p(a)^2]}{\mathbb{E}_a[\nabla \log p(a)^2]} \\
&= \frac{\mathbb{E}_a[(a+2)(a-\theta)^2]}{\mathbb{E}_a[(a-\theta)^2]} \\
&= \frac{\mathbb{E}_a[a^3 - 2a^2\theta + a\theta^2 + 2a^2 - 4a\theta + 2\theta^2]}{\mathbb{E}_a[a^2 - 2a\theta + \theta^2]} \\
&= \frac{\mathbb{E}_a[a^3] - 2\,\mathbb{E}_a[a^2]\theta + \mathbb{E}_a[a]\theta^2 + 2\,\mathbb{E}_a[a^2] - 4\,\mathbb{E}_a[a]\theta + 2\theta^2}{\mathbb{E}_a[a^2] - 2\,\mathbb{E}_a[a]\theta + \theta^2} \\
&= \frac{\mathbb{E}_a[a^3] + 2(1-\theta)\,\mathbb{E}_a[a^2] + \theta^3 - 4\theta^2 + 2\theta^2}{\mathbb{E}_a[a^2] - 2\theta^2 + \theta^2} \\
&= \frac{(\theta^3 + 3\theta) + 2(1-\theta)(1+\theta^2) + \theta^3 - 4\theta^2 + 2\theta^2}{(1+\theta^2) - 2\theta^2 + \theta^2} \\
&= \theta^3 + 3\theta + 2 + 2\theta^2 - 2\theta - 2\theta^3 + \theta^3 - 4\theta^2 + 2\theta^2 \\
&= \theta + 2
\end{aligned}
$$

6.1.3

$$
\begin{aligned}
\mathop{\mathbb{E}}_{\tau}\left[\sum_{t=1}^{T} \nabla \log \pi(a_t|s_t) b(s_t)\right] &= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{\tau}\left[\nabla \log \pi(a_t|s_t) b(s_t)\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{\tau_t}\left[\nabla \log \pi(a_t|s_t) b(s_t)\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t,a_t}\left[\nabla \log \pi(a_t|s_t) b(s_t)\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t}\left[\sum_{a_t} \pi(a_t|s_t) \nabla \log \pi(a_t|s_t) b(s_t)\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t}\left[b(s_t) \sum_{a_t} \pi(a_t|s_t) \frac{\nabla \pi(a_t|s_t)}{\pi(a_t|s_t)}\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t}\left[b(s_t) \sum_{a_t} \nabla \pi(a_t|s_t)\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t}\left[b(s_t) \nabla \sum_{a_t} \pi(a_t|s_t)\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t}\left[b(s_t) \nabla 1\right] \\
&= \sum_{t=1}^{T} \mathop{\mathbb{E}}_{s_t}\left[b(s_t) 0\right] \\
&= 0
\end{aligned}
$$

6.2 Compatible Function Approximation Theorem

6.2.1

Without loss of generality, we can focus on only on particular state s. Since we never use any property of this state, the proof holds for any state.

$$\begin{aligned}
\mathop{\mathbb{E}}_{a}[\hat{q}_{\boldsymbol{w}}(s,a)] &= \mathop{\mathbb{E}}_{a}[\boldsymbol{w}^T \nabla_\theta \log \pi_\theta(s,a)] \\
&= \boldsymbol{w}^T \mathop{\mathbb{E}}_{a}[\nabla_\theta \log \pi_\theta(s,a)] \\
&= \boldsymbol{w}^T \sum_a \pi_\theta(s,a) \nabla_\theta \log \pi_\theta(s,a) \\
&= \boldsymbol{w}^T \sum_a \pi_\theta(s,a) \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)} \\
&= \boldsymbol{w}^T \sum_a \nabla_\theta \pi_\theta(s,a) \\
&= \boldsymbol{w}^T \nabla_\theta \sum_a \pi_\theta(s,a) \\
&= \boldsymbol{w}^T \nabla_\theta 1 \\
&= \boldsymbol{w}^T 0 \\
&= 0
\end{aligned}$$

Since the expectation over the actions of the approximated q function $\hat{q}_{\boldsymbol{w}}(s,a)$ is always 0, it is not an unbiased estimate of the real q function.

6.2.2

$$\begin{aligned}
\mathop{\mathbb{E}}_{a}[q_\pi(s,a) - v_\pi(s)] &= \mathop{\mathbb{E}}_{a}[q_\pi(s,a)] - v_\pi(s) \\
&= v_\pi(s) - v_\pi(s) \\
&= 0
\end{aligned}$$

which means: $\mathbb{E}_a[A(s,a)] = 0$.

6.2.3

$\hat{q}_{\boldsymbol{w}}(s,a)$ can be interpreted more like an approximation of the *advantage* function $A(s,a)$ than of the q-function q. Indeed, the expectation of $\hat{q}_{\boldsymbol{w}}(s,a)$ is not $v_\pi(s)$ (as the expectation of q) but 0 (as the expectation of the advantage).

6.2.4

$$
\begin{aligned}
\nabla_\theta \log \pi_\theta(a,s) &= \frac{1}{\pi_\theta(a,s)} \nabla_\theta \pi_\theta(a,s) \\
&= \frac{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}}{e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}}} \nabla_\theta \frac{e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}}}{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}} \\
&= \frac{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}}{e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}}} \frac{\left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right) \nabla_\theta e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}} - e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}} \nabla_\theta \left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right)}{\left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right)^2} \\
&= \frac{1}{e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}}} \frac{\left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right) e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}} \boldsymbol{\phi}_{s,a} - e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}} \left(\sum_b \nabla_\theta e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right)}{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}} \\
&= \frac{1}{e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}}} \frac{\left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right) e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}} \boldsymbol{\phi}_{s,a} - e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,a}} \left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}} \boldsymbol{\phi}_{s,b}\right)}{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}} \\
&= \frac{\left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}\right) \boldsymbol{\phi}_{s,a} - \left(\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}} \boldsymbol{\phi}_{s,b}\right)}{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}} \\
&= \boldsymbol{\phi}_{s,a} - \sum_b \frac{e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}} \boldsymbol{\phi}_{s,b}}{\sum_b e^{\boldsymbol{\theta}^T \boldsymbol{\phi}_{s,b}}} \\
&= \boldsymbol{\phi}_{s,a} - \sum_b \pi_\theta(s,b) \boldsymbol{\phi}_{s,b}
\end{aligned}
$$

Then:

$$
\begin{aligned}
\hat{q}_{\boldsymbol{w}}(s,a) &= \boldsymbol{w}^T \left[\nabla_\theta \log \pi_\theta(a,s)\right] \\
&= \boldsymbol{w}^T \left[\boldsymbol{\phi}_{s,a} - \sum_b \pi_\theta(s,b) \boldsymbol{\phi}_{s,b}\right]
\end{aligned}
$$

6.3 Natural Gradient

6.3.1

$$
\begin{aligned}
\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \mu} &= \frac{\partial \left(\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right)\right)}{\partial \mu} \\
&= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right) \frac{\partial \left(-\frac{(a-\mu)^2}{2\sigma^2}\right)}{\partial \mu} \\
&= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right) \frac{-1}{2\sigma^2} \frac{\partial (a-\mu)^2}{\partial \mu} \\
&= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right) \frac{-1}{2\sigma^2} 2(\mu - a) \\
&= \frac{a-\mu}{\sigma^2} \pi(a|\mu,\sigma^2)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \sigma} &= \frac{\partial \left(\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) \right)}{\partial \sigma} \\
&= \frac{\partial \left(\frac{1}{\sigma} \right)}{\partial \theta_\sigma} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) + \frac{1}{\sigma\sqrt{2\pi}} \frac{\partial \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right)}{\partial \sigma} \\
&= -\frac{1}{\sigma^2} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) - \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) (a-\mu)^2 \frac{1}{2} \frac{\partial \left(\frac{1}{\sigma^2} \right)}{\partial \sigma} \\
&= -\frac{1}{\sigma^2} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) - \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) (a-\mu)^2 \frac{1}{2} \frac{-2}{\sigma^3} \\
&= -\frac{1}{\sigma^2} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2} \right) - \frac{1}{\sigma\sqrt{2\pi}} \exp\left(+\frac{(a-\mu)^2}{2\sigma^2} \right) \frac{(a-\mu)^2}{\sigma^2} \frac{1}{\sigma} \\
&= \frac{1}{\sigma} \pi(a|\mu, \sigma^2) \left(\frac{(a-\mu)^2}{\sigma^2} - 1 \right)
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \theta_\mu} &= \frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \mu} \frac{\partial \mu}{\partial \theta_\mu} \\
&= \frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \mu} 1 \\
&= \frac{a-\mu}{\sigma^2} \pi(a|\boldsymbol{\theta}) \\
&= \frac{a-\theta_\mu}{\exp(2\theta_\sigma)} \pi(a|\boldsymbol{\theta})
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \theta_\sigma} &= \frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \sigma} \frac{\partial \sigma}{\partial \theta_\sigma} \\
&= \frac{1}{\exp(\theta_\sigma)} \pi(a|\boldsymbol{\theta}) \left(\frac{(a-\mu)^2}{\exp(2\theta_\sigma)} - 1 \right) \frac{\partial \sigma}{\partial \theta_\sigma} \\
&= \frac{1}{\exp(\theta_\sigma)} \pi(a|\boldsymbol{\theta}) \left(\frac{(a-\mu)^2}{\exp(2\theta_\sigma)} - 1 \right) \exp(\theta_\sigma) \\
&= \pi(a|\boldsymbol{\theta}) \left(\frac{(a-\mu)^2}{\exp(2\theta_\sigma)} - 1 \right)
\end{aligned}
$$

Finally:

$$
\begin{aligned}
\frac{\partial \log \pi(a|\boldsymbol{\theta})}{\partial \theta_\mu} &= \frac{1}{\pi(a|\boldsymbol{\theta})} \frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \theta_\mu} = \frac{a-\theta_\mu}{\exp(2\theta_\sigma)} \\
\frac{\partial \log \pi(a|\boldsymbol{\theta})}{\partial \theta_\sigma} &= \frac{1}{\pi(a|\boldsymbol{\theta})} \frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \theta_\sigma} = \frac{(a-\mu)^2}{\exp(2\theta_\sigma)} - 1
\end{aligned}
$$

6.3.2

Natural Policy Gradient produces a gradient for the update which is *covariant*, i.e. independent from the parametrization used. As a result, the gradient on different parameters is adapted to their scale and the correlation between parameters is taken into account. Conversely, in vanilla policy gradient, all parameters are considered independent and equally important; hence, in this case, the update steps depend on the particular parametrization used for the policy.

Moreover, by using Natural Policy Gradient, the optimization is done directly on the policy space instead of the parameter space and it always guarantees an improvement of the policy.

6.3.3

$$
\begin{aligned}
F_{\boldsymbol{\theta}} &= \mathop{\mathbb{E}}_{\tau} \left[\nabla_{d\boldsymbol{\theta}} \log \pi(a|\boldsymbol{\theta}_0 + d\boldsymbol{\theta}) \nabla_{d\boldsymbol{\theta}} \log \pi(a|\boldsymbol{\theta}_0 + d\boldsymbol{\theta})^T \right] \\
&= \mathop{\mathbb{E}}_{a \sim \pi_{\boldsymbol{\theta}}} \left[\nabla_{d\boldsymbol{\theta}} \log \pi(a|\boldsymbol{\theta}_0 + d\boldsymbol{\theta}) \nabla_{d\boldsymbol{\theta}} \log \pi(a|\boldsymbol{\theta}_0 + d\boldsymbol{\theta})^T \right] \\
&= \mathop{\mathbb{E}}_{a \sim \pi_{\boldsymbol{\theta}}} \left[\nabla_{\boldsymbol{\theta}_0} \log \pi(a|\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}_0} \log \pi(a|\boldsymbol{\theta}_0)^T \right]
\end{aligned}
$$

Let's consider an entry of F at the time. We index the vector $\nabla_{\boldsymbol{\theta}_0} \log \pi(a|\boldsymbol{\theta}_0)$ with 0 (corresponding to the gradient wrt θ_μ) and 1 (corresponding to the gradient wrt θ_σ). We index F accordingly, as outer product of that vector with itself. So:

$$
\begin{aligned}
F_{00} &= \mathop{\mathbb{E}}_{a \sim \pi_{\boldsymbol{\theta}}} \left[\nabla_{\theta_{0\mu}} \log \pi(a|\boldsymbol{\theta}_0)^2 \right] \\
&= \mathop{\mathbb{E}}_{a \sim \pi_{\boldsymbol{\theta}}} \left[\left(\frac{a - \theta_{0\mu}}{\exp(2\theta_{0\sigma})} \right)^2 \right] \\
&= \frac{1}{\exp(2\theta_{0\sigma})^2} \mathop{\mathbb{E}}_{a \sim \pi_{\boldsymbol{\theta}}} \left[(a - \theta_{0\mu})^2 \right] \\
&= \frac{1}{\exp(2\theta_{0\sigma})^2} \mathop{\mathbb{E}}_{a \sim \mathcal{N}(a|\theta_{0\mu}, \exp(2\theta_{0\sigma}))} \left[(a - \theta_{0\mu})^2 \right] \quad \text{notice the second central moment of a Gaussian distribution} \\
&= \frac{1}{\exp(2\theta_{0\sigma})^2} \exp(2\theta_{0\sigma}) \\
&= \frac{1}{\exp(2\theta_{0\sigma})} \\
&= \frac{1}{\sigma(\theta_{0\sigma})^2}
\end{aligned}
$$

$$
\begin{aligned}
F_{10} = F_{01} &= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\nabla_{\theta_{0\mu}} \log \pi(a|\boldsymbol{\theta}_0) \nabla_{\theta_{0\sigma}} \log \pi(a|\boldsymbol{\theta}_0)\right] \\
&= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{a-\theta_{0\mu}}{\exp(2\theta_{0\sigma})}\left(\frac{(a-\theta_{0\mu})^2}{\exp(2\theta_{0\sigma})} - 1\right)\right] \\
&= \frac{1}{\exp(2\theta_{0\sigma})} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})(a-\theta_{0\mu})^2}{\exp(2\theta_{0\sigma})} - (a-\theta_{0\mu})\right] \\
&= \frac{1}{\exp(2\theta_{0\sigma})} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})^3}{\exp(2\theta_{0\sigma})} - a + \theta_{0\mu}\right] \\
&= \frac{1}{\exp(2\theta_{0\sigma})}\left(\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})^3}{\exp(2\theta_{0\sigma})}\right] - \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}[a] + \theta_{0\mu}\right) \\
&= \frac{1}{\exp(2\theta_{0\sigma})}\left(\frac{\mathbb{E}_{a\sim\pi_{\boldsymbol{\theta}}}[(a-\theta_{0\mu})^3]}{\exp(2\theta_{0\sigma})} - \theta_{0\mu} + \theta_{0\mu}\right) \\
&= \frac{1}{\exp(2\theta_{0\sigma})^2}\left(\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}[(a-\theta_{0\mu})^3]\right) \qquad \text{third central momentum of a Gaussian} \\
&= \frac{1}{\exp(2\theta_{0\sigma})^2} 0 \\
&= 0
\end{aligned}
$$

$$
\begin{aligned}
F_{11} &= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\nabla_{\theta_{0\sigma}} \log \pi(a|\boldsymbol{\theta}_0)^2\right] \\
&= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\left(\frac{(a-\theta_{0\mu})^2}{\exp(2\theta_{0\sigma})} - 1\right)^2\right] \\
&= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})^4}{\exp(2\theta_{0\sigma})^2}\right] - 2 \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})^2}{\exp(2\theta_{0\sigma})}\right] + 1 \\
&= \frac{1}{\exp(2\theta_{0\sigma})^2} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[(a-\theta_{0\mu})^4\right] - 2\frac{1}{\exp(2\theta_{0\sigma})} \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[(a-\theta_{0\mu})^2\right] + 1 \\
&= \frac{3\exp(2\theta_{0\sigma})^2}{\exp(2\theta_{0\sigma})^2} - 2\frac{\exp(2\theta_{0\sigma})}{\exp(2\theta_{0\sigma})} + 1 \\
&= 3 - 2 + 1 \\
&= 2
\end{aligned}
$$

Therefore: $F = \begin{bmatrix} \frac{1}{\sigma(\theta_{0\sigma})^2} & 0 \\ 0 & 2 \end{bmatrix}$.

6.3.4

We can notice that F is a diagonal matrix. As a result, according to this metric, the two parameters are still independent from each other but are just scaled according to their corresponding weight on the diagonal.

Moreover, since the inverse of a diagonal matrix is the matrix with the inverses of the elements in the diagonal, it means that the update rule will use the original gradient with each dimension "normalized" by the corresponding weight in the diagonal of F, which can be interpreted as the scale of the parameter. We can also notice that the scale of θ_σ is always constant while the scale of θ_μ depends on the value of the variance. Indeed, using the update rule, the update to θ_μ becomes larger as the variance increases but it shrinks when the variance is low, which is what one would expect.

6.3.5

Notice that, since the parametrization of μ is the same, $\nabla_{\theta_\mu} \log \pi$ and $F_{0,0}$ are unchanged (substituting $\sigma(\theta_\sigma)$ with θ_σ).

We have shown before that:

$$\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \sigma} = \frac{1}{\sigma}\pi(a|\mu,\sigma^2)\left(\frac{(a-\mu)^2}{\sigma^2} - 1\right)$$

Therefore:

$$\begin{aligned}
\frac{\partial \log \pi(a|\boldsymbol{\theta})}{\partial \theta_\sigma} &= \frac{1}{\pi(a|\boldsymbol{\theta})}\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \theta_\sigma} \\
&= \frac{1}{\pi(a|\boldsymbol{\theta})}\frac{\partial \pi(a|\boldsymbol{\theta})}{\partial \sigma(\theta_\sigma)}\frac{\partial \sigma(\theta_\sigma)}{\partial \theta_\sigma} \\
&= \frac{1}{\pi(a|\boldsymbol{\theta})}\frac{1}{\sigma(\theta_\sigma)}\pi(a|\boldsymbol{\theta})\left(\frac{(a-\mu)^2}{\sigma(\theta_\sigma)^2} - 1\right)\frac{\partial \sigma(\theta_\sigma)}{\partial \theta_\sigma} \\
&= \frac{1}{\pi(a|\boldsymbol{\theta})}\frac{1}{\sigma(\theta_\sigma)}\pi(a|\boldsymbol{\theta})\left(\frac{(a-\mu)^2}{\sigma(\theta_\sigma)^2} - 1\right)1 \\
&= \frac{1}{\sigma(\theta_\sigma)}\left(\frac{(a-\mu)^2}{\sigma(\theta_\sigma)^2} - 1\right) \\
&= \frac{1}{\theta_\sigma}\left(\frac{(a-\mu)^2}{\theta_\sigma^2} - 1\right)
\end{aligned}$$

Then:

$$\begin{aligned}
F_{10} = F_{01} &= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\nabla_{\theta_{0\mu}} \log \pi(a|\boldsymbol{\theta}_0)\nabla_{\theta_{0\sigma}} \log \pi(a|\boldsymbol{\theta}_0)\right] \\
&= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{a-\theta_{0\mu}}{\theta_{0\sigma}^2}\frac{1}{\theta_{0\sigma}}\left(\frac{(a-\theta_{0\mu})^2}{\theta_{0\sigma}^2} - 1\right)\right] \\
&= \frac{1}{\theta_{0\sigma}^3}\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})^3}{\theta_{0\sigma}^2} - a + \theta_{0\mu}\right] \\
&= \frac{1}{\theta_{0\sigma}^5}\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[(a-\theta_{0\mu})^3\right] && \text{third central momentum of a Gaussian} \\
&= \frac{1}{\theta_{0\sigma}^5}0 \\
&= 0
\end{aligned}$$

$$
\begin{aligned}
F_{11} &= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\nabla_{\theta_{0\sigma}}\log\pi(a|\boldsymbol{\theta}_0)^2\right] \\
&= \mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{1}{\theta_{0\sigma}^2}\left(\frac{(a-\theta_{0\mu})^2}{\theta_{0\sigma}^2}-1\right)^2\right] \\
&= \frac{1}{\theta_{0\sigma}^2}\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}\left[\frac{(a-\theta_{0\mu})^4}{\theta_{0\sigma}^4}-2\frac{(a-\theta_{0\mu})^2}{\theta_{0\sigma}^2}+1\right] \\
&= \frac{1}{\theta_{0\sigma}^2}\left(\frac{1}{\theta_{0\sigma}^4}\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}[(a-\theta_{0\mu})^4]-\frac{2}{\theta_{0\sigma}^2}\mathop{\mathbb{E}}_{a\sim\pi_{\boldsymbol{\theta}}}[(a-\theta_{0\mu})^2]+1\right) \\
&= \frac{1}{\theta_{0\sigma}^2}\left(\frac{1}{\theta_{0\sigma}^4}3\theta_{0\sigma}^4-\frac{2}{\theta_{0\sigma}^2}\theta_{0\sigma}^2+1\right) \\
&= \frac{2}{\theta_{0\sigma}^2}
\end{aligned}
$$

Therefore: $F = \begin{bmatrix} \frac{1}{\theta_{0\sigma}^2} & 0 \\ 0 & \frac{2}{\theta_{0\sigma}^2} \end{bmatrix}$.

6.3.6

In the second case, the "scale" for θ_σ is not constant but depends on the value of the parameter. As a result, updates are larger when the variance parameter is larger, while the updates become smaller when the variance is closer to 0. Anyways, this behaviour is similar to the previous parameterization where $\sigma = \exp(\theta_\sigma)$: indeed, in that case, though the scale of θ_σ is always constant, the corresponding update to the resulting σ becomes smaller as θ_σ shrinks (and, so, σ comes closer to 0), while, when θ_σ is large, small updates to this parameter corresponds to larger updates to σ. This means that, independently of the parametrization used, the update steps on the resulting σ are similar.

6.3.7

First, notice that $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = G(\tau)\nabla_{\boldsymbol{\theta}}\log\pi(a|\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}\log\pi(a|\boldsymbol{\theta})$.

Now, using the first parametrization, $\sigma = \exp(\theta_\sigma)$; therefore: $\theta_{0\sigma} = \ln 4$ and $\theta_{0\mu} = 0$. Then:

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) &= \nabla_{\boldsymbol{\theta}}\log\pi(a|\boldsymbol{\theta}_0) \\
&= \begin{pmatrix} \frac{a-\theta_{0\mu}}{\exp(2\theta_{0\sigma})} \\ \frac{(a-\theta_{0\mu})^2}{\exp(2\theta_{0\sigma})}-1 \end{pmatrix} \\
&= \begin{pmatrix} \frac{8-0}{4^2} \\ \frac{(8-0)^2}{4^2}-1 \end{pmatrix} \\
&= \begin{pmatrix} 1/2 \\ 3 \end{pmatrix}
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
&\text{Vanilla PG} && \boldsymbol{\theta}^* = \boldsymbol{\theta}_0 + \alpha\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) = \begin{pmatrix} 0 \\ \ln 4 \end{pmatrix} + 0.01\begin{pmatrix} 1/2 \\ 3 \end{pmatrix} = \begin{pmatrix} 0.005 \\ 1.416294361 \end{pmatrix} \\
&\text{Natural PG} && \boldsymbol{\theta}^* = \boldsymbol{\theta}_0 + \alpha F^{-1}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) = \begin{pmatrix} 0 \\ \ln 4 \end{pmatrix} + 0.01\begin{bmatrix} 4^2 & 0 \\ 0 & 1/2 \end{bmatrix}\begin{pmatrix} 1/2 \\ 3 \end{pmatrix} = \begin{pmatrix} 0.08 \\ 1.401294361 \end{pmatrix}
\end{aligned}
$$

The resulting values of σ are:

Vanilla PG	$\sigma = 4.121818135$
Natural PG	$\sigma = 4.060452258$

Instead, using the second parametrization, $\sigma = \theta_\sigma$; therefore: $\theta_{0\sigma} = 4$ and $\theta_{0\mu} = 0$. Then:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) &= \nabla_{\boldsymbol{\theta}} \log \pi(a|\boldsymbol{\theta}_0) \\
&= \begin{pmatrix} \frac{a-\theta_{0\mu}}{\theta_{0\sigma}^2} \\ \frac{1}{\theta_{0\sigma}} \left(\frac{(a-\theta_{0\mu})^2}{\theta_{0\sigma}^2} - 1 \right) \end{pmatrix} \\
&= \begin{pmatrix} \frac{8-0}{4^2} \\ \frac{1}{4} \left(\frac{(8-0)^2}{4^2} - 1 \right) \end{pmatrix} \\
&= \begin{pmatrix} 1/2 \\ 3/4 \end{pmatrix}
\end{aligned}$$

Therefore:

$$\text{Vanilla PG} \qquad \boldsymbol{\theta}^* = \boldsymbol{\theta}_0 + \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) = \begin{pmatrix} 0 \\ 4 \end{pmatrix} + 0.01 \begin{pmatrix} 1/2 \\ 3/4 \end{pmatrix} = \begin{pmatrix} 0.005 \\ 4.0075 \end{pmatrix}$$

$$\text{Natural PG} \qquad \boldsymbol{\theta}^* = \boldsymbol{\theta}_0 + \alpha F^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) = \begin{pmatrix} 0 \\ 4 \end{pmatrix} + 0.01 \begin{bmatrix} 4^2 & 0 \\ 0 & 4^2/2 \end{bmatrix} \begin{pmatrix} 1/2 \\ 3/4 \end{pmatrix} = \begin{pmatrix} 0.08 \\ 4.06 \end{pmatrix}$$

The resulting values of σ are:

Vanilla PG	$\sigma = 4.0075$
Natural PG	$\sigma = 4.06$

In conclusion, we can notice that using the Vanilla Policy Gradient leads to two different values for σ using the two different parameterizations. Conversely, as expected, when using Natural Policy Gradient the resulting values of σ after the update rule are very similar for both parametrizations. This is, indeed, the property of Natural Policy Gradient: the result of the update step on the model is covariant and does not depend on the actual parametrization used for it.