Reinforcement Learning

Assignment 2 (Theoretical Questions)

Group Members

Group ID - 31
Matheus da Silva Araujo – 261218407
Miguel Ángel Carrillo – 261205372

February 8, 2026

1 Theoretical questions

1.1 Question 1

Consider a discounted Markov Decision Process (MDP) where the rewards are bounded:

$$|R(s,a)| \leq R_{\max} \quad \text{for all states } s \in \mathcal{S} \text{ and actions } a \in \mathcal{A}$$

The value of state s for the policy π is defined as:

$$v_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=t}^{\infty} \gamma^{k-t} R_{k+1} \,\middle|\, S_t = s \right]$$

What is the maximum value of $v_\pi(s)$ across all states and policies? Provide an upper bound on $|v_\pi(s)|$ in terms of $R_{\max}$ and γ.

Answer:

We start from the definition of the state-value function:

$$v_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=t}^{\infty} \gamma^{k-t} R_{k+1} \,\middle|\, S_t = s \right], \quad \gamma \in [0,1)$$

Assume that rewards are bounded:

$$|R(s,a)| \leq R_{\max} \quad \forall s \in \mathcal{S},\ a \in \mathcal{A}$$

We first bound the absolute value of the return:

$$\left|\sum_{k=t}^{\infty}\gamma^{k-t}R_{k+1}\right| \leq \sum_{k=t}^{\infty}\gamma^{k-t}|R_{k+1}| \leq \sum_{k=t}^{\infty}\gamma^{k-t}R_{\max}$$

Re-indexing with $n = k - t$, we obtain:

$$\sum_{k=t}^{\infty}\gamma^{k-t}R_{\max} = R_{\max}\sum_{n=0}^{\infty}\gamma^{n}$$

If $0 \leq \gamma < 1$, the geometric series converges:

$$\sum_{n=0}^{\infty}\gamma^{n} = \frac{1}{1-\gamma}$$

If we substitute into our earlier inequality:

$$\left|\sum_{k=t}^{\infty}\gamma^{k-t}R_{k+1}\right| \leq \frac{R_{\max}}{1-\gamma}$$

Now we substitute this bound into the value function and use the triangle inequality $|\mathbb{E}[X]| \leq \mathbb{E}[|X|]$:

$$|v_\pi(s)| = \left|\mathbb{E}_\pi\left[\sum_{k=t}^{\infty}\gamma^{k-t}R_{k+1} \;\middle|\; S_t = s\right]\right| \leq \mathbb{E}_\pi\left[\left|\sum_{k=t}^{\infty}\gamma^{k-t}R_{k+1}\right| \;\middle|\; S_t = s\right] \leq \frac{R_{\max}}{1-\gamma}$$

Finally, provided that $\gamma \in [0, 1)$ so that the discounted return is not infinite, we have:

$$\boxed{|v_\pi(s)| \leq \frac{R_{\max}}{1-\gamma} \quad \forall s,\ \forall \pi}$$

1.2 Question 2

Define the λ-return G_t^λ and the corresponding Bellman operator T^λ Prove that T^λ is a contraction mapping in the L_∞ norm. In other words, show that:

$$\left\|T^\lambda v - T^\lambda u\right\|_\infty \leq \eta \left\|v - u\right\|_\infty$$

for some $\eta < 1$ Briefly explain the rate of contraction as a function of λ and γ.

Answer:

We first define the λ-return and its Bellman operator. Then we show that this operator is a contraction in the L_∞ norm.

Definition of the λ-return. For a value function v, the n-step return is

$$G_t^{(n)}(v) = \sum_{k=0}^{n-1} \gamma^k R_{t+k+1} + \gamma^n v(S_{t+n})$$

The λ-return is defined as a weighted average of all n-step returns:

$$G_t^{\lambda}(v) = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}(v)$$

Here, $\lambda \in [0, 1)$ controls how much weight is given to longer returns.

Bellman operator $\mathcal{T}^{\lambda}$. The Bellman operator associated with the λ-return is

$$(\mathcal{T}^{\lambda} v)(s) = \mathbb{E}_{\pi}\Big[G_t^{\lambda}(v) \mid S_t = s\Big]$$

This operator takes a value function v and returns the expected λ-return starting from state s.

Using linearity of expectation, we can rewrite this operator as

$$\mathcal{T}^{\lambda} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} \mathcal{T}^{(n)}$$

where $\mathcal{T}^{(n)}$ is the Bellman operator based on the n-step return.

Step 1: Contraction of the n-step Bellman operator. We first show that for any two value functions u and v,

$$\|\mathcal{T}^{(n)} v - \mathcal{T}^{(n)} u\|_{\infty} \leq \gamma^n \|v - u\|_{\infty}$$

By definition, the n-step Bellman operator applied to v is

$$(\mathcal{T}^{(n)} v)(s) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{n-1} \gamma^k R_{t+k+1} + \gamma^n v(S_{t+n}) \;\middle|\; S_t = s\right]$$

and a similar expression holds for $(\mathcal{T}^{(n)} u)(s)$.

We subtract the two expressions. The reward terms are exactly the same in both cases, so they cancel out. This leaves

$$(\mathcal{T}^{(n)} v)(s) - (\mathcal{T}^{(n)} u)(s) = \mathbb{E}_{\pi}\big[\gamma^n \big(v(S_{t+n}) - u(S_{t+n})\big) \;\big|\; S_t = s\big]$$

Next, we take absolute values and use the inequality $|\mathbb{E}[X]| \leq \mathbb{E}[|X|]$:

$$\Big|(\mathcal{T}^{(n)} v)(s) - (\mathcal{T}^{(n)} u)(s)\Big| \leq \mathbb{E}_{\pi}[\gamma^n \, |v(S_{t+n}) - u(S_{t+n})| \mid S_t = s]$$

By definition of the L_{∞} norm,

$$|v(x) - u(x)| \leq \|v - u\|_{\infty} \quad \text{for any state } x.$$

Applying this to the random state S_{t+n} gives

$$\left|(\mathcal{T}^{(n)}v)(s) - (\mathcal{T}^{(n)}u)(s)\right| \leq \gamma^n \|v-u\|_\infty$$

Finally, taking the maximum over all states s, we obtain

$$\|\mathcal{T}^{(n)}v - \mathcal{T}^{(n)}u\|_\infty \leq \gamma^n \|v-u\|_\infty$$

Step 2: Contraction of $\mathcal{T}^\lambda$. Using the expression of $\mathcal{T}^\lambda$ as a weighted sum of n-step operators, we write

$$\mathcal{T}^\lambda v - \mathcal{T}^\lambda u = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1}\big(\mathcal{T}^{(n)}v - \mathcal{T}^{(n)}u\big)$$

We now take the L_∞ norm and apply the triangle inequality:

$$\|\mathcal{T}^\lambda v - \mathcal{T}^\lambda u\|_\infty \leq (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1}\|\mathcal{T}^{(n)}v - \mathcal{T}^{(n)}u\|_\infty$$

Substituting the bound from the previous step gives

$$\|\mathcal{T}^\lambda v - \mathcal{T}^\lambda u\|_\infty \leq (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1}\gamma^n\|v-u\|_\infty$$

Step 3: Contraction rate. Let

$$\eta = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1}\gamma^n$$

This sum is a geometric series and can be computed as

$$\sum_{n=1}^{\infty} \lambda^{n-1}\gamma^n = \gamma\sum_{m=0}^{\infty}(\lambda\gamma)^m = \frac{\gamma}{1-\lambda\gamma},$$

which is valid since $\gamma < 1$ and $\lambda \in [0,1)$.

Therefore,

$$\eta = \frac{\gamma(1-\lambda)}{1-\lambda\gamma}$$

We conclude that

$$\|\mathcal{T}^\lambda v - \mathcal{T}^\lambda u\|_\infty \leq \eta\|v-u\|_\infty$$

with $\eta < 1$ for $\gamma < 1$. This shows that $\mathcal{T}^\lambda$ is a contraction mapping in the L_∞ norm.

Interpretation The contraction rate η depends on both the discount factor γ and the parameter λ. The discount factor γ determines how strongly future values influence the current estimate, while λ controls how much weight is given to longer returns.

Smaller values of λ place more emphasis on short, bootstrapped returns. As a result, the operator contracts more strongly and convergence is faster. Larger values of λ place more weight on long returns, which reduces the amount of bootstrapping and weakens the contraction.

In particular, when $\lambda = 0$, the λ-return reduces to the one-step TD target, and the contraction rate becomes $\eta = \gamma$, which corresponds to TD(0). As $\lambda \to 1$, the contraction rate approaches $\eta \to 1$, meaning that the operator becomes only weakly contractive. This case corresponds to Monte Carlo methods, which rely on full returns and do not strongly shrink errors at each update.

1.3 Question 3

(a) Explain the bias–variance trade-off between the Monte Carlo first-visit and every-visit Monte Carlo algorithms. What happens to the bias asymptotically?

Both algorithms are unbiased. This is because both of them use the "true" return as a target for learning. Asymptotically, as the number of episodes goes to infinity, both methods converge to the true value function, and the bias goes to zero.

The main difference can be found in the variance. Every-visit Monte Carlo uses all occurrences of a state within an episode, which introduces correlated samples and can lead to higher variance. First-visit Monte Carlo uses only the first occurrence of a state per episode, reducing correlation between updates and typically resulting in lower variance.

(b) Discuss bias-variance trade-off between first-visit Monte Carlo and TD(0) learning algorithms.

TD(0) introduces bias because it uses bootstrapping: the target includes the current value estimate rather than the true return. In contrast, Monte Carlo methods use the full return as the target and are therefore unbiased. In the tabular case, the bias of TD(0) vanishes asymptotically as the value function converges.

In terms of variance, Monte Carlo methods typically have higher variance because they rely on complete returns, which depend on all future rewards and can vary significantly across episodes. TD(0) reduces variance by using a one-step target and bootstrapping from existing estimates, leading to faster and more stable learning.

(c) Consider state aggregation for value function approximation as discussed in the class. Explain the bias-variance trade-off as a function of the number of aggregates to estimate the value function. How does the quality of estimates affect when you just have a handful of returns to approximate the value function.

Aggregating states reduces the number of parameters to be estimated by grouping multiple states into a single aggregate. When the number of aggregates is small, many states share the same value estimate, which increases bias because differences between states within the same aggregate cannot be represented. However, this sharing of data reduces variance, since each aggregate is updated using returns from multiple states.

As the number of aggregates increases, the approximation becomes more expressive and the bias decreases, but the variance increases because fewer samples are available per aggregate.

When only a handful of returns are available, using fewer aggregates often leads to better-quality estimate because, even if the bias is higher, the estimates have lower variance and are

more stable. With many aggregates and limited data, estimates can have very high variance and be unreliable.

2 Author Contribution

All the team members contributed equally to the assignment. Matheus lead the coding questions, while Miguel lead the theory questions. Subsequently, Matheus contributed to verify correctness and fix errors in the theory part, while Miguel did the same for the coding exploration questions.

3 LLM Usage Statement

For the theory questions, ChatGPT was used to fix LaTeX compilation errors, e.g., *The following block of equations is not compiling on Overleaf, how to fix it?*. No LLM usage for the coding part.