Deep Reinforcement Learning (Sp25)

Instructor: Dr. Mohammad Hossein Rohban

Quiz Solutions - Lecture 24

Solution By: Arshia Gharooni

Q1

Which of the following statements correctly contrasts a KL-divergence constraint with a support (MMD) constraint in offline RL?

1. A KL constraint forces the learned policy to match the behaviour policy everywhere, whereas a support constraint only requires their action distributions to share support and therefore allows greater deviation on in-support actions.
2. A KL constraint is less conservative than a support constraint because it ignores state–action pairs never seen in the data.
3. A support (MMD) constraint penalises differences in means and variances only, whereas KL penalises the entire distribution.
4. Both constraints are equivalent in tabular MDPs; differences arise only with deep networks.

Correct Answers: A

A KL constraint (e.g., $D_{\mathrm{KL}}(\pi \,\|\, \mu)$) penalises any deviation from the behaviour policy and becomes infinite if π puts mass outside $\mathrm{supp}(\mu)$, keeping π close *everywhere*. A support/MMD constraint primarily enforces shared support (stay within the data support), allowing larger reweighting among in-support actions. (B)–(D) are incorrect.

Q2

In the context of offline Q-learning with function approximation, bootstrapping-error amplification refers to

1. The accumulation of Monte-Carlo estimation noise when rewards are highly stochastic.
2. The geometric growth of value-function over-estimation caused by repeatedly backing-up Q-values for state–action pairs that lie outside the data set's support.
3. The divergence that appears when the actor network is not updated as frequently as the critic.
4. A numerical instability arising from using large discount factors with expectile regression.

Correct Answers: B

In offline Q-learning, targets come from Bellman backups

$$\mathcal{T}Q(s,a) = r + \gamma \max_{a'} Q(s',a').$$

When $(s',a') \notin \text{supp}(\mathcal{D})$, a function approximator extrapolates with error ε. Repeated bootstrapping propagates and compounds these extrapolation errors roughly as

$$\varepsilon,\ \gamma\varepsilon,\ \gamma^2\varepsilon,\ \dots$$

leading to geometric growth (amplification) and over-estimation outside the dataset support. Options A, C, and D describe different issues (Monte-Carlo noise, actor–critic scheduling, and expectile-specific numerics) and are not the bootstrapping-error phenomenon.

Q3

In the KL-constrained formulation of Offline RL, explain in one or two sentences how the optimal policy update turns into AWR.

Imposing a per-state KL constraint $\mathrm{KL}(\pi(\cdot|s) \,\|\, \mu(\cdot|s)) \leq \epsilon$ with behavior policy μ yields the Lagrangian optimum

$$\pi^*(a|s) \;\propto\; \mu(a|s)\, \exp\!\Big(\tfrac{A(s,a)}{\tau}\Big),$$

where $A(s,a) = Q(s,a) - V(s)$ and $\tau > 0$ is the temperature (inverse of the KL multiplier). Projecting this nonparametric π^* onto a parametric π_θ by minimizing $\mathrm{KL}(\pi^* \| \pi_\theta)$ gives the weighted behavior-cloning update

$$\theta \leftarrow \arg\max_\theta \; \mathbb{E}_{(s,a)\sim\mathcal{D}}\Big[\exp\Big(\tfrac{A(s,a)}{\tau}\Big)\, \log \pi_\theta(a|s)\Big],$$

which is exactly the Advantage-Weighted Regression (AWR) objective.

Q4

Why does *bootstrapping-error amplification* occur in ordinary Q-learning when applied to a fixed offline dataset, and what is one principled algorithmic modification that removes or reduces this problem?

In offline Q-learning, the TD target

$$r + \gamma \max_{a'} Q_\theta(s', a')$$

queries Q_θ for actions a' that may be *out of the dataset's support*. Function-approximation errors on these OOD actions are *selected by the max* (overestimation) and then *bootstrapped* into future targets without any new data to correct them, so the error grows (amplifies) across iterations.

One principled fix (CQL). *Conservative Q-Learning (CQL)* adds a conservatism regularizer that pushes down values for unsupported actions while keeping data actions high, e.g. add to the TD loss

$$\alpha \left(\mathbb{E}_{s\sim D}\big[\log \sum_a e^{Q_\theta(s,a)} \big] - \mathbb{E}_{(s,a)\sim D}\big[Q_\theta(s,a) \big] \right),$$

which penalizes large Q on actions not seen in D. This reduces OOD overestimation and thus mitigates bootstrapping-error amplification.