Deep Reinforcement Learning (Sp25)

Instructor: Dr. Mohammad Hossein Rohban

Summary of Lecture 17: Policy-based Theoretical Guarantees

Summarized By: Arshia Gharooni

Let an episodic Markov decision process (MDP) be given by the tuple

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma, \rho_0),$$

where $\mathcal{S}$ and $\mathcal{A}$ are measurable state- and action-spaces, $P(s'|s,a)$ is the transition kernel, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the (possibly stochastic) reward, $0 < \gamma < 1$ is the discount factor and ρ_0 the initial-state distribution. For any stationary, stochastic policy $\pi_\theta(a|s)$ with parameters $\theta \in \mathbb{R}^d$ we write

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\Big[\textstyle\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\Big], \qquad \eta_\theta(s) = \textstyle\sum_{t=0}^{\infty} \gamma^t \Pr_{\pi_\theta}[s_t = s]$$

for its expected return and its discounted state-occupancy measure, respectively. The action-value and state-value functions are

$$Q_{\pi_\theta}(s,a) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, s_0 = s,\, a_0 = a\right], \qquad V_{\pi_\theta}(s) = \mathbb{E}_{a\sim\pi_\theta}[Q_{\pi_\theta}(s,a)].$$

Throughout, gradients are taken with respect to the policy parameters unless stated otherwise.

Recap: The Policy-Gradient Theorem

The classical policy-gradient theorem asserts

$$\nabla_\theta J(\theta) = \mathbb{E}_{s\sim\eta_\theta,\, a\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s)\, Q_{\pi_\theta}(s,a)]\,.$$

Because the expectation is taken under the state distribution η_θ induced *by the very same policy* being optimised, the estimator remains unbiased even when trajectories are gathered on-policy. In practice, one uses the variance-reduced form

$$\nabla_\theta J(\theta) = \mathbb{E}_{s,a}\Big[\nabla_\theta \log \pi_\theta(a|s)\, \underbrace{\big(Q_{\pi_\theta}(s,a) - b(s)\big)}_{A_{\pi_\theta}(s,a)}\Big],$$

where $b(s)$ is any baseline independent of a. Choosing $b(s) = V_{\pi_\theta}(s)$ yields the advantage function $A_{\pi_\theta}(s,a)$.

Policy Gradient as Generalised Policy Iteration

Policy iteration alternates *policy evaluation* and *greedy improvement*. A first-order algorithm that performs *one* gradient-based improvement step per evaluation round may likewise be interpreted as a *soft* variant of policy iteration:

1. **Evaluation step.** Estimate Q_{π_k} or A_{π_k} for the current policy π_k.
2. **Improvement step.** Update parameters according to

$$\theta_{k+1} = \theta_k + \alpha_k\, \nabla_\theta J(\theta)\big|_{\theta=\theta_k}.$$

The following proposition formalises the intuitive claim that, for sufficiently small step-size, a policy-gradient step realises policy improvement.

Proposition 18.1 (Guaranteed Improvement under Step-Size Constraint). *Let $L(\theta;\theta_k) = J(\theta_k) + \nabla J(\theta_k)^\top(\theta - \theta_k)$ be the local linearisation of the objective. If θ_{k+1} satisfies*

$$D_{\mathrm{KL}}\big(\pi_{\theta_k} \,\|\, \pi_{\theta_{k+1}}\big) \;\le\; \frac{2(1-\gamma)}{C}\, \alpha_k^2$$

for a constant $C > 0$ bounding the advantage function, then $J(\theta_{k+1}) \geq J(\theta_k)$.

Proof. The performance-difference lemma gives

$$J(\theta_{k+1}) - J(\theta_k) = \frac{1}{1-\gamma}\, \mathbb{E}_{s\sim\eta_{\theta_{k+1}},\, a\sim\pi_{\theta_{k+1}}}\big[A_{\pi_{\theta_k}}(s,a)\big].$$

Replacing $\eta_{\theta_{k+1}}$ by η_{θ_k} introduces a distribution-mismatch error bounded via

$$\big|\eta_{\theta_{k+1}}(s) - \eta_{\theta_k}(s)\big| \;\leq\; \frac{\gamma}{1-\gamma} \max_{s'} D_{\mathrm{TV}}\big(\pi_{\theta_{k+1}}(\cdot|s') \,\|\, \pi_{\theta_k}(\cdot|s')\big).$$

Pinsker's inequality relates total-variation and Kullback–Leibler divergences, producing a second-order penalty in the step-size. Collecting terms yields the claimed sufficient condition. □

Bounding Distribution Shift

Because policies are parametrised continuously, successive iterates differ only slightly. Let

$$\bar{D} = \max_s D_{\mathrm{TV}}\big(\pi_{\theta_{k+1}}(\cdot|s) \,\|\, \pi_{\theta_k}(\cdot|s)\big).$$

Then one obtains the occupancy-measure perturbation bound

$$\big\|\eta_{\theta_{k+1}} - \eta_{\theta_k}\big\|_1 \;\leq\; \frac{\gamma}{(1-\gamma)^2}\, \bar{D}.$$

Consequently, the performance difference decomposes into

$$\underbrace{\frac{1}{1-\gamma}\, \mathbb{E}_{\eta_{\theta_k}, \pi_{\theta_{k+1}}}\big[A_{\pi_{\theta_k}}(s,a)\big]}_{\text{optimistic local model}} \;-\; \underbrace{\frac{4\gamma}{(1-\gamma)^2} R_{\max} \bar{D}}_{\text{penalty}}.$$

The explicit penalty term motivates either constraining $\bar{D}$ (TRPO) or augmenting the objective with a soft regulariser (PPO).

From Hard to Soft Policy Iteration: The Maximum-Entropy Principle

Classical RL seeks a deterministic optimal policy. The *maximum-entropy* framework augments the return with an entropy bonus:

$$J_{\text{soft}}(\theta) = \mathbb{E}_\tau\Big[\sum_{t=0}^{\infty} \gamma^t \big(r(s_t,a_t) + \alpha\, \mathcal{H}\big(\pi_\theta(\cdot|s_t)\big)\big)\Big],$$

where $\alpha > 0$ controls the exploration–exploitation trade-off. The associated soft-Q-function satisfies the *soft Bellman equation*

$$Q^\star(s,a) = r(s,a) + \gamma\, \mathbb{E}_{s'}\big[V^\star(s')\big], \quad V^\star(s) = \alpha \log \int_{\mathcal{A}} \exp\Big(\tfrac{1}{\alpha} Q^\star(s,a')\Big)\, da'.$$

The Soft-Actor-Critic (SAC) Algorithm

Critic Update

Given experience replay buffer $\mathcal{D}$, minimise the soft Bellman residual

$$\mathcal{L}_Q(\psi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\Big[\big(Q_\psi(s,a) - \hat{y}(r,s')\big)^2\Big],$$

$$\hat{y}(r,s') = r + \gamma\, \mathbb{E}_{a'\sim\pi_\theta(\cdot|s')}\big[Q_{\bar{\psi}}(s',a') - \alpha \log \pi_\theta(a'|s')\big],$$

with a slowly moving target network $Q_{\bar{\psi}}$. Under standard conditions, fixed-point iteration on this objective converges to the soft optimal $Q^\star$.

Actor Update

The policy parameters are updated by one step of *information projection*,

$$\nabla_\theta J_{\text{soft}}(\theta) = \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi_\theta}\big[\nabla_\theta \log \pi_\theta(a|s)\,\big(\alpha \log \pi_\theta(a|s) - Q_\psi(s,a)\big)\big].$$

Equivalently, π_θ is the solution of

$$\min_{\pi\in\Pi} \mathbb{E}_{s\sim\mathcal{D}}\big[D_{\mathrm{KL}}\big(\pi(\cdot|s)\,\|\,\exp\big((Q_\psi(s,\cdot) - c_s)/\alpha\big)\big)\big],$$

with log-partition term c_s ensuring normalisation. In practice, the gradient is estimated using the reparametrisation trick: for Gaussian policies $\pi_\theta(\cdot|s) = \mathcal{N}(\mu_\theta(s), \Sigma_\theta(s))$ one writes $a = \mu_\theta(s) + \Sigma_\theta^{1/2}(s)\,\epsilon$, $\epsilon \sim \mathcal{N}(0, I)$.

Temperature Adaptation

The entropy-temperature α can itself be treated as a learnable parameter with objective

$$\mathcal{L}_\alpha(\alpha) = \mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\big[-\alpha\,\big(\log \pi_\theta(a|s) + \bar{\mathcal{H}}\big)\big],$$

driving the expected entropy toward a user-specified target $\bar{\mathcal{H}}$. Gradient descent on α preserves the monotonically increasing nature of J_{soft}.

Soft Policy Evaluation, Improvement and Iteration

Soft Policy Evaluation iteratively applies the soft Bellman operator

$$\mathcal{T}^\pi_{\text{soft}} Q = r + \gamma\,\mathbb{E}_{s',a'\sim\pi}\big[Q(s',a') - \alpha \log \pi(a'|s')\big].$$

The operator is a contraction in the sup-norm with modulus γ, guaranteeing unique fixed-point Q_π.

Soft Policy Improvement. Given Q_π, construct

$$\pi_{\text{new}}(\cdot|s) \;\propto\; \exp\Big(\tfrac{1}{\alpha}\,Q_\pi(s,\cdot)\Big),$$

which is *provably* better in the soft-return sense: $J_{\text{soft}}(\pi_{\text{new}}) \geq J_{\text{soft}}(\pi)$.

Soft Policy Iteration alternates evaluation and improvement, converging to a policy that maximises the maximum-entropy objective. SAC instantiates an *approximate* version wherein only a single gradient step is taken in each stage.

Loss-Function Summary

- **Critic:** $\mathcal{L}_Q(\psi) = \frac{1}{2}\,(Q_\psi - \hat{y})^2$.
- **Actor:** $\mathcal{L}_\pi(\theta) = \mathbb{E}_{s\sim\mathcal{D},\,\epsilon\sim\mathcal{N}}\big[\alpha \log \pi_\theta(a_\theta(s,\epsilon)|s) - Q_\psi(s, a_\theta(s,\epsilon))\big]$.
- **Temperature:** $\mathcal{L}_\alpha(\alpha) = -\alpha\,(\log \pi_\theta(a|s) + \bar{\mathcal{H}})$.

Gradient noise is tempered by large-batch replay; target networks and Polyak averaging further stabilise training.