Online Learning in Kernelized Markov Decision Processes

Sayak Ray Chowdhury Aditya Gopalan

Department of Electrical Communication Engineering, Indian Institute of Science

Problem Statement

Episodically maximize reward in an *unknown* Markov Decision Process $M = \{\mathcal{S}, \mathcal{A}, R, P, H\}$

- State space $\mathcal{S} \subseteq \mathbb{R}^m$, known
- Action space $\mathcal{A} \subseteq \mathbb{R}^n$, known
- Reward distribution $R(s,a)$, unknown
- Transition distribution $P(s,a)$, unknown
- Episode length H, known

Definitions

- Policy $\pi : \mathcal{S} \times \{1, \dots, H\} \to \mathcal{A}$
- *Finite horizon undiscounted* Value function $V_{\pi,h}(s) = \mathbb{E}[\sum_{j=h}^{H} \overline{R}(s_j, a_j) \mid s_h = s]$
- Optimal policy $\pi_\star \in \text{argmax}_\pi V_{\pi,h}(s) \;\; \forall s, \; \forall h$
- Agent choses policy π_l at episode l
- Cumulative Regret $= \sum_l \mathbb{E}[V_{\pi_\star,1}(s) - V_{\pi_l,1}(s)]$

At every round h within an episode, an agent:

1. Takes an action $a_h \in \mathcal{A}$ based on the current state $s_h \in \mathcal{S}$ and past observations
2. Receives reward $r_h \sim R(s_h, a_h)$
3. Observes next state $s_{h+1} \sim P(s_h, a_h)$

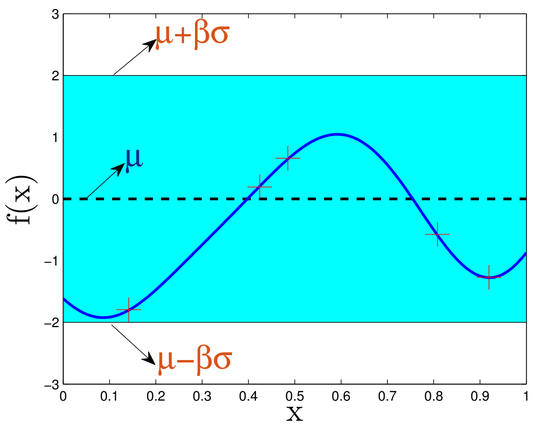
Assumptions

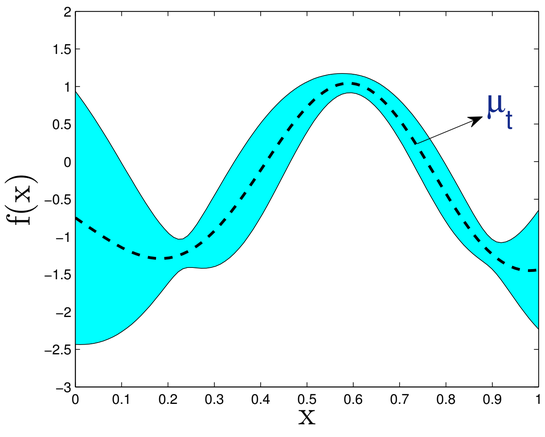
- Reward $r_h = \overline{R}(s_h, a_h) + \varepsilon_R$
- Next state $s_{h+1} = \overline{P}(s_h, a_h) + \varepsilon_P$
- $\overline{R}$, $\overline{P}$ elements of reproducing kernel Hilbert spaces
- ε_R and ε_P are samples of zero-mean, additive sub-Gaussian noise
- *One step future value function* is Lipschitz

Goal: Minimize the loss incurred in the *Value function* due to not knowing the optimal policy $\pi_\star$ and instead using any other policy π_l at episode l

Algorithm Design: How to Choose Policy?

Building Block: Gaussian Process (GP) prior and Gaussian likelihood model





- Represent uncertainty over any unknown function f using **G**aussian **p**rocess prior $GP(0, k(x,y))$
- Squared Exponential (SE) kernel: $k(x,y) = \exp\left(\frac{-\|x-y\|_2^2}{2l^2}\right)$
- Observe t reward samples $y = f(x) + \text{noise}$
- Noise: iid Gaussian $\mathcal{N}(0, \lambda)$

Posterior of f: $GP(\mu_t(x), k_t(x,y))$

$$\mu_t(x) = k_t(x)^T (K_t + \lambda I)^{-1} Y_t$$
$$k_t(x,y) = k(x,y) - k_t(x)^T (K_t + \lambda I)^{-1} k_t(y)$$

No Regret Algorithms: GP-UCRL and PSRL

Bayesian Inference Philosophy: Put *separate* Gaussian process priors over mean reward and mean transition function, and update posteriors at the end of every episode

1. Construct two confidence sets, one each for mean reward and mean transition function, using parameters of posterior distributions
2. Find the set of all MDPs, for which mean reward and mean transition function lie within respective confidence sets and choose the optimal policy for that *set of MDPs*

1. Sample two random functions, one each from the posterior distributions of mean reward and mean transition function
2. Build an MDP using the random samples of mean reward and mean transition function and choose the optimal policy for that *sampled MDP* (Osband et al., NeurIPS 2013)

Theorem 1: Cumulative Regret of **GP-UCRL** is $O\Big(\big(\gamma_T(R) + \gamma_{mT}(P)\big)\sqrt{T}\Big)$ *with high probability*

Theorem 2: Expected Cumulative Regret of **PSRL** is $O\Big(\big(\gamma_T(R) + \gamma_{mT}(P)\big)\sqrt{T}\Big)$

- $\gamma_t(P)$ (resp. $\gamma_t(R)$) roughly represents the *maximum information gain* about the unknown dynamics (resp. rewards) after t rounds – measure reduction in uncertainty
- polylog(t) for common kernels (e.g. Polynomial, Squared Exponential) and for their products and sums

Key Idea: *At every episode/round, the unknown mean reward and mean transition function lie within properly constructed confidence sets of shrinking width*

Application: Linear Quadratic Regulator (LQR) Control

- **Model:** $s_{h+1} = As_h + Ba_h + \varepsilon_P$ and $r_h = s_h^T P s_h + a_h^T Q a_h + \varepsilon_R$ (Abbasi-Yadkori et al., COLT 2011)
- A, B, P and Q are unknown matrices, P and Q positive-definite
- Linear kernel structure for state transitions and quadratic kernel structure for rewards

Corollary 1: Cumulative Regret of **GP-UCRL** is $O\Big(\big(m^2 + n^2 + m(m+n)\big)\sqrt{T}\Big)$ *with high probability*

Corollary 2: Expected Cumulative Regret of **PSRL** is $O\Big(\big(m^2 + n^2 + m(m+n)\big)\sqrt{T}\Big)$

Computational Challenges and Open Questions

1. GP-UCRL requires *optimistic planning over a family of* MDPs: generally not tractable
2. PSRL requires *optimal planning for only a single* MDP: Is it tractable for continuous state/action MDPs?
3. If not, can we design an approximate MDP planner for a single MDP?
4. If so, can we obtain (through extended value iteration or otherwise) an efficient approximate planner for a family of MDPs?

References

[1] *(More) efficient reinforcement learning via posterior sampling, I. Osband, D. Russo, and B. Van Roy, NeurIPS 2013.*

[2] *Regret bounds for the adaptive control of linear quadratic systems, Y. Abbasi-Yadkori and C. SzepesvÃąri, COLT 2011.*

Acknowledgment. Google India PhD fellowship grant in Machine Learning, 2017