A Kernel-Based Approach to Non-Stationary Reinforcement Learning in Metric Spaces

Omar D. Domingues[1], Pierre Ménard[1], Matteo Pirotta[2], Emilie Kaufmann[1,3], Michal Valko[1,4]

[1]Inria Lille – Nord Europe, France [2]Facebook AI Research [3]CNRS & Université de Lille, France [4]DeepMind

Theoretical Foundations of Reinforcement Learning Workshop, ICML 2020

July 17, 2020

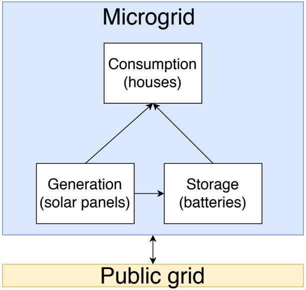
Motivation

Real word applications:

- Continuous states and actions
- Environment may change over time

Microgrid management:

- Electrical network with energy generation, storage and consumption
- Can buy energy from an external source (public grid)
- Goal: minimize cost

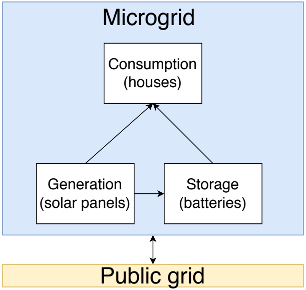


Motivation

State: [production, consumption, batteries] $\in \mathbb{R}^d$
Actions: [charge, discharge] the batteries
Reward: $-1 \times$(cost of generating/buying energy)

Non-stationary behavior:

- Fluctuations of the energy price
- New consumer in the network
- Degradation of batteries



Model

Episodic RL: K episodes of horizon H

Non-stationary MDP:

At time (k, h):

- State $x_h^k \in \mathcal{X}$
- Action $a_h^k \in \mathcal{A}$
- Reward $\widetilde{r}_h^k \in [0,1]$ with mean $r_h^k(x_h^k, a_h^k)$
- Next state $x_{h+1}^k \sim \mathrm{P}_h^k(\cdot | x_h^k, a_h^k)$

Optimal value functions in episode k:

$$\mathrm{Q}_{k,h}^*(x,a) = r_h^k(x,a) + \mathrm{P}_h^k \mathrm{V}_{k,h+1}^*(x,a)$$
$$\mathrm{V}_{k,h}^*(x) = \max_{a \in \mathcal{A}} \mathrm{Q}_{k,h}^*(x,a)$$

Goal & Regularity Assumptions

Goal: minimize the dynamic regret

$$\underbrace{\mathcal{R}(K)}_{\text{regret}} = \sum_{k=1}^{K} \left(\underbrace{V^*_{1,k}(x_1)}_{\text{value of an optimal agent}} - \underbrace{V^{\pi_k}_{1,k}(x_1)}_{\text{value of the algorithm}} \right)$$

Assumption: Lipschitz continuity w.r.t. a known metric ρ:

$$\left| r_h^k(x,a) - r_h^k(x',a') \right| \leq L_{\mathrm{r}} \rho \left[(x,a),(x',a') \right]$$

$$\underbrace{\mathbb{W}_1 \left(\mathrm{P}_h^k(\cdot|x,a), \mathrm{P}_h^k(\cdot|x',a') \right)}_{\text{1-Wasserstein distance}} \leq L_{\mathrm{p}} \rho \left[(x,a),(x',a') \right]$$

$\Longrightarrow$ optimal Q-functions are L-Lipschitz

Kernel-Based RL (KBRL)

Non-parametric model of the MDP:

$$\widehat{r}_h^k(x,a) = \frac{\sum_{s=1}^{k-1} w_h^{k,s}(x,a)\, \widetilde{r}_h^s}{\beta + \sum_{s=1}^{k-1} w_h^{k,s}(x,a)}, \quad \widehat{P}_h^k(y|x,a) = \frac{\sum_{s=1}^{k-1} w_h^{k,s}(x,a)\, \delta_{x_{h+1}^s}(y)}{\beta + \sum_{s=1}^{k-1} w_h^{k,s}(x,a)}$$

$w_h^{k,s}(x,a)$: similarity between (x,a) and (x_h^s, a_h^s) in episode k

- Depends on the distance $\rho\left[(x,a),(x_h^s,a_h^s)\right]$
- Depends on the time interval $(k-s)$

KBRL for stationary MDPs:

$\rightarrow$ First studied by [Ormoneit and Sen, 2002] (offline)

$\rightarrow$ Regret bounds provided by [Darwiche Domingues et al., 2020] (online)

Examples of similarity measures

$\{$sliding window, exponential discount$\} \times$ Gaussian kernel

sliding window [Garivier and Moulines, 2011, Gajane et al., 2018, Cheung et al., 2019]

$$w_h^{k,s}(x,a) = \mathbb{I}\{s \geq k - W\} \exp\left(-\frac{\rho[(x,a),(x_h^s,a_h^s)]^2}{2\sigma^2}\right)$$

discount [Kocsis and Szepesvári, 2006, Garivier and Moulines, 2011, Russac et al., 2019]

$$w_h^{k,s}(x,a) = \eta^{k-s-1} \exp\left(-\frac{\rho[(x,a),(x_h^s,a_h^s)]^2}{2\sigma^2}\right)$$

Space- and Time-Dependent Kernels

Kernel function $\overline{\Gamma}_{\eta,W} : \mathbb{N} \times \mathbb{R} \to [0,1]$ such that:

$$w_h^{k,s}(x,a) = \overline{\Gamma}_{\eta,W}\left(\underbrace{k-s-1}_{\text{time interval}}, \underbrace{\rho\left[(x,a),(x_h^s,a_h^s)\right]/\sigma}_{\text{distance}}\right)$$

σ: kernel bandwidth, η: discount parameter, W: window parameter

Assumptions:

- $z \mapsto \overline{\Gamma}_{\eta,W}(t,z)$ is Lipschitz continuous
- $\overline{\Gamma}_{\eta,W}(t,z) \leq c_1 \exp(-z^2/2)$
- $\overline{\Gamma}_{\eta,W}(t,z) \leq c_2 \eta^t$ if $t \geq W$: $\rightarrow$ forget old data (bias)
- $\overline{\Gamma}_{\eta,W}(t,z) \geq G(z)\eta^t$ if $t < W$: $\rightarrow$ remember recent data (variance)

Algorithm

KeRNS

For all state-action pairs (x_h^s, a_h^s) that were seen before, compute:

$$\textbf{(``training'')} \quad \widetilde{Q}_h^k(x_h^s, a_h^s) = \widehat{r}_h^k(x_h^s, a_h^s) + \widehat{P}_h^k V_{h+1}^k(x_h^s, a_h^s) + \mathtt{B}_h^k(x_h^s, a_h^s)$$

For any new state-action pair (x, a), define:

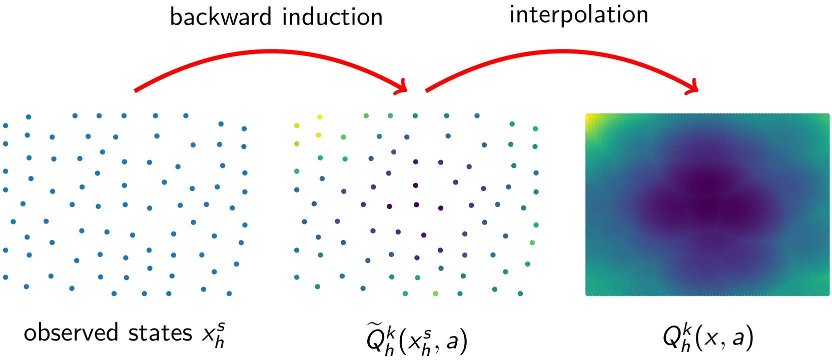
$$\textbf{(generalizing)} \quad Q_h^k(x, a) \stackrel{\text{def}}{=} \min_{s \in [k-1]} \left[\widetilde{Q}_h^k(x_h^s, a_h^s) + L\rho\left[(x, a), (x_h^s, a_h^s)\right] \right]$$

Play the policy: **(acting)** $\pi^k(h, x) \in \operatorname{argmax}_a Q_h^k(x, a)$

Exploration Bonus: $\mathtt{B}_h^k(x, a) \approx H/\sqrt{\mathbf{C}_h^k(x, a)} + \beta H/\mathbf{C}_h^k(x, a) + L\sigma$

$$\mathbf{C}_h^k(x, a) \stackrel{\text{def}}{=} \beta + \sum_{s=1}^{k-1} w_h^{k,s}(x, a) \approx \text{ number of recent visits to } (x, a)$$

Algorithm


backward induction
interpolation
observed states x_h^s
$\widetilde{Q}_h^k(x_h^s, a)$
$Q_h^k(x, a)$

Regret Bound

Variation of the MDP: $\Delta = \Delta^{\mathrm{r}} + L\Delta^{\mathrm{p}}$, where

$$\Delta^{\mathrm{r}} = \sum_{i,h} \sup_{x,a} \left| r_h^i(x,a) - r_h^{i+1}(x,a) \right|, \quad \Delta^{\mathrm{p}} = \sum_{i,h} \sup_{x,a} \mathbb{W}_1 \left(\mathrm{P}_h^i(\cdot|x,a), \mathrm{P}_h^{i+1}(\cdot|x,a) \right)$$

With probability at least $1-\delta$, the regret of KeRNS is bounded by $\approx$

$$\underbrace{H^2 K \sqrt{\log\left(\frac{1}{\eta}\right)} \sqrt{|\mathcal{C}'_\sigma| \, |\mathcal{C}_\sigma|}}_{\text{sum of exploration bonuses}} + \underbrace{HW\Delta + \frac{\eta^W}{1-\eta} KH^3}_{\text{bias due to changes}} + \underbrace{LKH\sigma}_{\text{bias due to smoothing}}$$

$\rightarrow |\mathcal{C}_\sigma|, |\mathcal{C}'_\sigma|$: σ-covering numbers of $\mathcal{X} \times \mathcal{A}$ and $\mathcal{X}$

Regret Bound

Tuning the kernel parameters to minimize the regret:

$$\log(1/\eta) = \Delta^{\frac{2}{3}} K^{-\frac{2d+2}{2d+3}}, \quad W = \log_\eta((1-\eta)/K), \quad \sigma = K^{-\frac{1}{2d+3}}$$

gives a regret bound

$$\mathcal{R}(K) \lesssim H^2 \Delta^{\frac{1}{3}} K^{\frac{2d+2}{2d+3}}$$

where $d =$ covering dimension of $\mathcal{X} \times \mathcal{A}$.

Requirement for sublinear regret: $\Delta < K^{3/(2d+3)}$

$\rightarrow$ Another analysis gives a bound $\mathcal{R}(K) \lesssim H^2 \Delta^{\frac{1}{2}} K^{\frac{2d+1}{2d+2}}$ (for $d > 0$)

$\rightarrow$ Optimal choice of parameters requires an upper bound on Δ

$\rightarrow$ For $d = 0$, matches the lower bound for bandits in terms of Δ and K [Besbes et al., 2014]

Conclusion & Perspectives

+ dynamic regret bounds for continuous non-stationary MDPs
+ flexible and easy to implement
+ weak assumptions on the MDP

– backward induction is slow
– curse of dimensionality

◦ lower bounds for $d > 0$?
◦ what if we don't know (an upper bound on) Δ?
◦ can we use these kernel-based bonuses in deep RL?

Thank you!

Questions?

Barreto, A. M., Precup, D., and Pineau, J. (2016). Practical kernel-based reinforcement learning. The Journal of Machine Learning Research, 17(1):2372–2441.

Besbes, O., Gur, Y., and Zeevi, A. (2014).
Stochastic multi-armed-bandit problem with non-stationary rewards.
In Advances in neural information processing systems, pages 199–207.

Cheung, W. C., Simchi-Levi, D., and Zhu, R. (2019).
Non-Stationary Reinforcement Learning: The Blessing of (More) Optimism.
arXiv e-prints, page arXiv:1906.02922.

Darwiche Domingues, O., Ménard, P., Pirotta, M., Kaufmann, E., and Valko, M. (2020).
Regret Bounds for Kernel-Based Reinforcement Learning.
arXiv e-prints, page arXiv:2004.05599.

Gajane, P., Ortner, R., and Auer, P. (2018).
A sliding-window algorithm for markov decision processes with arbitrarily changing rewards and transitions.
arXiv preprint arXiv:1805.10066.

Garivier, A. and Moulines, E. (2011).
On Upper-Confidence Bound Policies For Switching Bandit Problems.
In Algorithmic Learning Theory (ALT), pages 174–188. PMLR.

Kocsis, L. and Szepesvári, C. (2006).
Discounted UCB.
In 2nd PASCAL Challenges Workshop.

Kveton, B. and Theocharous, G. (2012).
Kernel-based reinforcement learning on representative states.
In Twenty-Sixth AAAI Conference on Artificial Intelligence.

Ormoneit, D. and Sen, Ś. (2002).
Kernel-based reinforcement learning.
Machine Learning, 49(2):161–178.

Russac, Y., Vernade, C., and Cappé, O. (2019).
Weighted linear bandits for non-stationary environments.
In Advances in Neural Information Processing Systems, pages 12017–12026.

Runtime

KeRNS requires a backward induction over (x_h^s, a_h^s) for $s = 1, \ldots, k-1$
$\implies \mathcal{O}\left(k^2\right)$ time per episode k

Idea: use m representative state-action pairs (x_i, a_i) instead!
[Kveton and Theocharous, 2012, Barreto et al., 2016]
$\implies$ constant time per episode: $\mathcal{O}\left(m^2\right)$

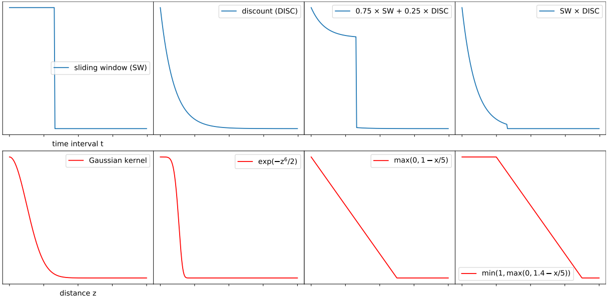
Impact on the regret: additive term $(\varepsilon/\sigma)H^3 K$, where

$$\varepsilon = \max_{s,h} \min_i \rho\left[(x_h^s, a_h^s), (x_i, a_i)\right]$$

and assuming a Gaussian kernel.

Examples of kernels

Simplest examples: time-dependent kernel $\times$ space-dependent kernel

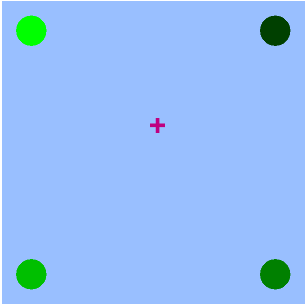


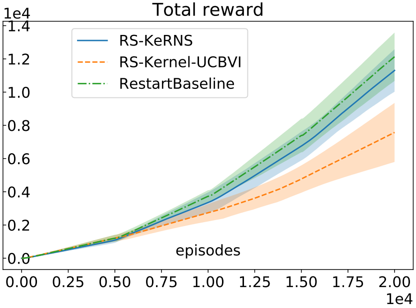
More generally, the kernel design can be adapted to the problem, e.g. :

- If the MDP changes only in certain regions, we can choose a kernel that forgets data only in those regions.
- If we know that a task is appearing periodically, we might use a periodic time-dependent kernel.

Experiments

`RS-KeRNS` = `KeRNS` on Representative States





Environment: continuous GridWorld, rewards change in the corners.