Learning to Optimize via Information-Directed Sampling

Russo and Van Roy [3]

0.1 Abstract

- Information-directed sampling: minimizes ratio between expected single-period regret and mutual information
- Expected regret bound for general class of models, scales with entropy of optimal action distribution
- SOA performance on Bernoulli/Gaussian/linear bandit problems

0.2 Introduction

- IDS can dramatically outperform UCB, TS, and knowledge-gradient algorithms
- IDS is more a heuristic than a specific algorithm; needs to be adapted to specific cases

0.3 Literature review

- Two previous papers have examined using mutual information between optimal action and next observation to guide action selection; [2, 4]. Both seem more focused on optimization of expensive-to-evaluate functions in low-dimensional continuous action spaces with a Gaussian process prior over the objective.
- Knowledge-gradient uses different measure of information; the impact of an observation on the quality of the decision made by a greedy algorithm. Can work well in some settings but no general guarantees and may not converge to optimality even in simple settings.
- Most literature focuses on contexts where goal is to converge to an optimal action in a way that limits exploration costs; not geared towards problems where time preference is important. KG is an exception; can take a discount factor as input for time preference. Some heuristics discussed for discounted problem, and recent work generalises TS to address discounted problems.
- Regret bounds built on information-theoretic analysis of TS, which bound regret of policy in terms of information ratio. Information ratio of IDS is always smaller than of TS so bounds on TS yield regret bounds for IDS.

0.4 Problem formulation

- Actions $(A_t)_{t\in\mathbb{N}}$ chosen from *finite* action set $\mathcal{A}$, outcomes $(Y_{t,A_t})_{t\in\mathbb{N}}$ observed
- Let $Y_t \equiv (Y_{t,a})_{a\in\mathcal{A}}$ be vector of outcomes at $t \in \mathbb{N}$
- There is some rv θ such that conditioned on θ, $(Y_t)_{t\in\mathbb{N}}$ is iid
- Agent associates a reward $R(y)$ with each outcome $y \in \mathcal{Y}$ where reward function $R : \mathcal{Y} \to \mathbb{R}$ is fixed and known. Denote by $R_{t,a} = R(Y_{t,a})$ the realized reward of action a at time t.
- Uncertainty in θ induces uncertainty about the true optimal action $A^* \in \operatorname{argmax}_{a\in\mathcal{A}} \mathbb{E}[R_{1,a} \mid \theta]$.
- T-period regret of actions $A_1, \ldots, A_T$ is

$$\text{Regret}(T) := \sum_{t=1}^{T} (R_{t,A^*} - R_{t,A_t});$$

here we study expected regret

$$\mathbb{E}[\text{Regret}(T)] = \mathbb{E}\left[\sum_{t=1}^{T} (R_{t,A^*} - R_{t,A_t})\right]$$

where expectation is taken over randomness in A_t, outcomes Y_t, and over the prior distribution over θ, i.e. we study the Bayesian regret

- Action A_t is chosen based on history $\mathcal{F}_t = (A_1, Y_{1,A_1}, \ldots, A_{t-1}, Y_{t-1,A_{t-1}})$. With slight abuse of notation write π_t for the distribution over actions at time t, i.e. $\pi_t(a) = \mathbb{P}(A_t = a \mid \mathcal{F}_t)$.

Further notation and information theory:

- Set $\alpha_t(a) = \mathbb{P}(A^* = a \mid \mathcal{F}_t)$ as the posterior distribution of A^*.
- For probability measures P and Q on a common measurable space, if P is absolutely continuous wrt Q (i.e. whenever A is Q-measurable, $Q(A) = 0$ implies $P(A) = 0$) the KL divergence between P and Q is
$$D_{\mathrm{KL}}(P \,||\, Q) = \int \log \frac{\mathrm{d}P}{\mathrm{d}Q} \mathrm{d}P$$
where $\frac{\mathrm{d}P}{\mathrm{d}Q}$ is the Radon-Nikodym derivative of P wrt Q.
- For a probability distribution P on a finite set $\mathcal{X}$, the Shannon entropy is
$$H(P) \coloneqq -\sum_{x \in \mathcal{X}} P(x) \log(P(x)).$$
- The mutual information under the posterior distribution between rvs $X_1 : \Omega \to \mathcal{X}_1$ and $X_2 : \Omega \to \mathcal{X}_2$ is
$$I_t(X_1; X_2) \coloneqq D_{\mathrm{KL}}(\mathbb{P}((X_1, X_2) \in \cdot \mid \mathcal{F}_t) \,||\, \mathbb{P}(X_1 \in \cdot \mid \mathcal{F}_t)\mathbb{P}(X_2 \in \cdot \mid \mathcal{F}_t)),$$
the KL divergence between the joint posterior of X_1 and X_2 and the product of the marginals. Note $I_t(X_1; X_2)$ is an rv because it depends on the conditional probability measure $\mathbb{P}(\cdot \mid \mathcal{F}_t)$.
- Define the information gain from action a as
$$g_t(a) = I_t(A^*; Y_{t,a}).$$
Can be shown this is equal to expected reduction in entropy of the posterior distribution of A^* due to observing $Y_t(a)$, i.e.
$$g_t(a) = \mathbb{E}[H(\alpha_t) - H(\alpha_{t+1}) \mid \mathcal{F}_t, A_t = a]$$
which is used a lot in this paper.
- Let $\Delta_t(a) \coloneqq \mathbb{E}[R_{t,A^*} - R_{t,a} \mid \mathcal{F}_t]$ denote the expected instantaneous regret of action a at time t.
- Overload g_t and Δ_t: for an action-sampling distribution π,
$$g_t(\pi) \coloneqq \sum_{a \in \mathcal{A}} \pi(a) g_t(a)$$
denotes expected information gain when actions are selected according to π.
$$\Delta_t(\pi) = \sum_{a \in \mathcal{A}} \pi(a) \Delta_t(a)$$
is defined analogously.
- Occasionally write $\mathbb{E}_t[\cdot] = \mathbb{E}_t[\cdot \mid \mathcal{F}_t]$ for conditional expectations under the posterior, and similarly write $\mathbb{P}_t(\cdot) = \mathbb{P}(\cdot \mid \mathcal{F}_t)$.

0.5 Algorithm design principles

0.5.1 Motivation

- Goal: minimize expected regret *over a time horizon* T, which will be large. For large T and moderate times $t \ll T$, mapping from belief state to action prescribed by a Bayes-optimal policy does not vary significantly between time steps, so can restrict attention to stationary heuristic policies.
- IDS designed to account for kinds of information including:
  - Indirect information (feedback about other actions even if there is no useful feedback about the selected action)
  - Cumulating information (can select an action to obtain feedback that does not immediately enable higher expected reward but can do when combined with feedback from subsequent actions)
  - Avoids irrelevant information

0.5.2 Information-directed sampling

- Balances obtaining low expected regret in current period and acquiring information. In particular
$$\pi_t^{\mathrm{IDS}} \in \operatorname{argmin}_{\pi \in \mathcal{D}(\mathcal{A})} \left\{ \Psi_t(\pi) := \frac{\Delta_t(\pi)^2}{g_t(\pi)} \right\}$$
where we call $\Psi_t(\pi)$ the *information ratio* of an action-sampling distribution π.
- Stationary randomized policy: randomized in that each action is randomly sampled from a distribution, and stationary in that this action distribution is determined by the posterior distribution of θ and is otherwise independent of the time period.

Randomization plays a fundamental role:

Example 0.1 – A known standard

- Two actions $\mathcal{A} = \{a_1, a_2\}$. Rewards from a_1 are $\mathrm{Ber}(1/2)$, rewards from a_2 are either $\mathrm{Ber}(3/4)$ with prior probability p_0 and $\mathrm{Ber}(1/4)$ with prior probability $1 - p_0$.
- Consider stationary deterministic policies. Then each action A_t is a deterministic function of p_{t-1}, the posterior probability conditioned on observations made through period $t-1$. Suppose for some $p_0 > 0$, the policy selects $A_1 = a_1$. This is an uninformative action, so $p_t = p_0$ and $A_t = a_1$ for all t, so expected regret is linear in time. So for any deterministic stationary policy, there is some prior probability p_0 such that expected regret grows linearly with time.

Later establish a sublinear bound on expected regret of IDS, so randomization plays a fundamental role.

0.5.3 Alternative design principles

UCB and TS

Can perform well in many problems and provide strong theoretical guarantees. However, in some cases can perform very poorly relative to IDS. In both cases, UCB and TS restrict attention to sampling actions that have some chance of being optimal.

Example 0.2 – A revealing action

- Let $\mathcal{A} = \{0, \dots, K\}$ be $K+1$ actions and let θ be uniform from a finite set $\Theta = \{1, \dots, K\}$ of K possible values. Consider bandit feedback $Y_{t,a} = R_{t,a}$. Under θ, the reward of action a is
$$R_{t,a} = \begin{cases} 1 & \theta = a \\ 0 & \theta \neq a,\ a \neq 0 \\ \frac{1}{2\theta} & a = 0. \end{cases}$$
- Action 0 is known to never yields the maximal reward, so is never selected by TS or UCB. Instead, these algorithms select among actions $\{1, \dots, K\}$ ruling out only a single action at a time until a reward 1 is earned and the optimal action is found, so expected regret is linear in K.
- IDS recognizes that much more is learned by drawing action 0 than by selecting another action; selecting action 0 immediately identifies the optimal action.

Example 0.3 – Sparse linear model

- Linear bandit problem, $\mathcal{A} \subseteq \mathbb{R}^d$, reward from $a \in \mathcal{A}$ is $a^\intercal \theta^*$. True parameter θ drawn uniformly at random from set of 1-sparse vectors
$$\Theta = \left\{ \theta' \in \{0,1\}^d : \left\|\theta'\right\|_0 = 1 \right\}.$$
Assume $d = 2^m$ for some $m \in \mathbb{N}$. Action set is taken to be the set of vectors in $\{0,1\}^d$ normalized to be a unit vector in the L^1 norm, i.e.
$$\mathcal{A} = \left\{ \frac{x}{\|x\|_1} : x \in \{0,1\}^d,\ x \neq 0 \right\}.$$

- When an action a is selected and $y = a^\intercal \theta \in \{0, 1/\|a\|_0\}$ is observed, each $\theta' \in \Theta$ with $a^\intercal \theta' \neq y$ is ruled out. Let Θ_t be the set of parameters in Θ consistent with rewards observed up to time t, and let $\mathcal{I}_t = \{i \in \{1, \dots, d\} : \theta'_i = 1, \theta' \in \Theta_t\}$ denote the corresponding set of possible positive components.
- Note $A^* = \theta$, i.e. if θ were known we should choose it all the time. TS and UCB only choose actions from the support of A^* and therefore will only sample actions $a \in \mathcal{A}$ which, like A^*, have a single positive component. Unless that is also the positive component in θ, the algorithm gets a reward of 0 and rules out only one element of $\mathcal{I}_t$, so in the worst case we need d samples to identify the optimal action.
- IDS effectively performs binary search: it selects $a \in \mathcal{A}$ with $a_i > 0$ for half the components $i \in \mathcal{I}_t$ and $a_i = 0$ for the other half as well as any $i \notin \mathcal{I}_t$. After $\log_2(d)$ time steps the true value of θ is identified.
- Why? All params in Θ_t are equally likely, hence the expected reward of an action a is $\frac{1}{|\mathcal{I}_t|}\sum_{i \in I_t} a_i$. Since $a_i \geq 0$ and $\sum_i a_i = 1$ for each $a \in \mathcal{A}$, every action whose positive components are in $\mathcal{I}_t$ yields the highest possible expected reward of $1/|\mathcal{I}_t|$. Therefore binary search minimizes expected regret in period t for this problem. At the same time, binary search is assured to rule out half the parameter vectors in Θ_t at each time t. This is the largest possible expected reduction, and also leads to the largest possible information gain about A^*. Since binary search both minimizes expected regret in period t and uniquely maximizes expected information gain in period t, it is the sampling strategy followed by IDS.
- Here we can explicitly calculate the information ratio of each policy.
  - First consider Thompson sampling. The numerator in $\Psi_1(\pi)$ is $\Delta_1(\pi)^2$, and

    $$\Delta_1(\pi) = \sum_{a \in \mathcal{A}} \pi(a) \mathbb{E}[R_{1,A^*} - R_{1,a} \mid \mathcal{F}_1].$$

    Note that since $\mathcal{F}_t = (A_1, Y_{1,A_1}, \dots, A_{t-1}, Y_{t-1,A_{t-1}})$, $\mathcal{F}_1 = \varnothing$ is empty and conditioning on it has no effect.
    Since TS only chooses actions from the support of A^*, and there are d 1-sparse vectors, we have

    $$\pi_1^{\mathrm{TS}}(a) = \begin{cases} 0 & a \text{ is not 1-sparse} \\ 1/d & a \text{ is 1-sparse.} \end{cases}$$

    Rewards are deterministic, and we have $R_{1,A^*} = 1$, $R_{1,a} = \mathbb{1}_{a=A^*}$. Therefore

    $$\mathbb{E}[R_{1,A^*} - R_{1,a} \mid \mathcal{F}_1] = 1 - \mathbb{1}_{a=A^*} = \mathbb{1}_{a \neq A^*}.$$

    So we have

    $$\Delta_1(\pi_1^{TS}) = \sum_{\substack{a \text{ 1-sparse} \\ a \neq A^*}} \pi_1^{TS}(a) = (d-1) \cdot \frac{1}{d},$$

    so it follows that the numerator of $\Psi_1(\pi_1^{\mathrm{TS}}) = (d-1)^2/d^2$.
    The denominator in $\Psi_1(\pi_1^{\mathrm{TS}})$ is $g_1(\pi_1^{\mathrm{TS}}) = \sum_{a \in \mathcal{A}} \pi_1^{\mathrm{TS}}(a) g_1(a)$, where

    $$\begin{aligned} g_t(a) &= \mathbb{E}[H(\alpha_t) - H(\alpha_{t+1}) \mid \mathcal{F}_t, A_t = a] \\ \alpha_t(a) &= \mathbb{P}(A^* = a \mid \mathcal{F}_t). \end{aligned}$$

    Now $\pi_1^{\mathrm{TS}}(a)$ is as before. We have

    $$g_1(\pi_1^{\mathrm{TS}}) = \sum_{a \in \mathcal{A}} \pi_1^{\mathrm{TS}}(a) g_1(a) = \pi_1^{\mathrm{TS}}(A^*) g_1(A^*) + \sum_{a \neq A^*} \pi_1^{\mathrm{TS}}(a) g_1(a),$$

    and

    $$\begin{aligned} \pi_1^{\mathrm{TS}}(A^*) g_1(A^*) &= \frac{1}{d} g_1(A^*) \\ \sum_{a \neq A^*} \pi_1^{\mathrm{TS}}(a) g_1(a) &= \frac{d-1}{d} g_1(a), \quad a \neq A^*, \end{aligned}$$

since π_1^{TS} only assigns mass to those actions a which are 1-sparse.
By definition, $g_1(a) = \mathbb{E}[H(\alpha_1) - H(\alpha_2) \mid \mathcal{F}_1,\, A_1 = a]$. We start by finding $g_1(a)$ for $a \neq A^*$. Recall that $\mathcal{F}_1 = \varnothing$ so conditioning on it has no effect.

$$\alpha_1(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_1) = \begin{cases} \frac{1}{d} & a' \text{ is 1-sparse} \\ 0 & \text{otherwise,} \end{cases}$$

so $H(\alpha_1) = \log(d)$.
For α_2, note that $\mathcal{F}_2 = (A_1, Y_{1,A_1}) = (A_1, 0)$, since we are conditioning on $A_1 = a \neq A^*$. Consequently

$$\alpha_2(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_2) = \begin{cases} 0 & a' = A_1 \\ \frac{1}{d-1} & a' \neq A_1,\, a' \text{ is 1-sparse} \\ 0 & \text{otherwise} \end{cases}$$

so $H(\alpha_2) = \log(d-1)$.
Hence $H(\alpha_1) - H(\alpha_2) = \log\left(\frac{d}{d-1}\right)$, and we have $g_1(a') = \log\left(\frac{d}{d-1}\right)$ for $a' \neq A^*$.
For $a' = A^*$, note again that $\mathcal{F}_1 = \varnothing$ and therefore $H(\alpha_1) = \log d$ by the same reasoning as before. This time however $\mathcal{F}_2 = (A_1, Y_{1,A_1}) = (A^*, 1)$, since we condition on $A_1 = A^*$. Therefore

$$\alpha_2(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_2) = \begin{cases} 0 & a' \neq A_1 \\ 1 & a' = A_1 \end{cases}$$

so $H(\alpha_2) = 0$.
Finally we obtain

$$g_1(\pi_1^{\mathrm{TS}}) = \frac{\log d}{d} + \frac{d-1}{d}\log\left(\frac{d}{d-1}\right)$$

so that

$$\Psi_1(\pi_1^{\mathrm{TS}}) = \frac{(d-1)^2/d^2}{\frac{\log(d)}{d} + \frac{d-1}{d}\log\left(\frac{d}{d-1}\right)} \sim \frac{d}{\log(d)}.$$

– For IDS, first consider $\Delta_1(a) = \mathbb{E}[R_{1,A^*} - R_{1,a} \mid \mathcal{F}_1]$. Say a 1-sparse vector "aligns" with action a with non-negative entries if the 1 in the 1-sparse vector is in the same index as a positive entry in a.
Any action we choose at time step 1 has $d/2$ positions positive, so to be a unit vector in L^1-norm all positions which are positive must have value $2/d$. The reward $R_{1,a}$ is therefore $2/d$ if A^* aligns with a and is 0 if A^* does not align with a. These events happen for half of the values of a which can be chosen by IDS, and each value of a is chosen by IDS with equal probability. Therefore

$$\begin{aligned} \Delta_1(\pi_1^{IDS}) &= \sum_{a\colon A^* \text{ aligns with } a} \pi_1^{\mathrm{IDS}}(a)\left(1 - \frac{2}{d}\right) + \sum_{a\colon A^* \text{ does not align with } a} \pi_1^{\mathrm{IDS}}(a)\cdot 1 \\ &= \frac{1}{2}\left(1 - \frac{2}{d}\right) + \frac{1}{2}\cdot 1 = \left(1 - \frac{1}{d}\right), \end{aligned}$$

so that the numerator of $\Psi_1(\pi_1^{\mathrm{IDS}})$ is $(1 - 1/d)^2$.
Now consider

$$g_t(\pi) = \sum_a \pi_1(a) g_1(a) = \sum_{a \neq A^*} \pi_1(a) g_1(a),$$

where the last term follows because π_1^{IDS} picks actions a with half the components positive (which can never include the optimal action A^*). We have

$$\alpha_1(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_1) = \begin{cases} 1/d & a' \text{ is 1-sparse} \\ 0 & \text{otherwise,} \end{cases}$$

and $H(\alpha_1) = \log d$ as before. Now the first action we pick either aligns with $A*$ and gets reward $2/d$, with probability $1/2$, or gets a reward of 0 with probability $1/2$. So

$$\mathcal{F}_2 = (a, Y_{1,a}) = \begin{cases} (a,0) & \text{w.p. } 1/2 \\ (a, 2/d) & \text{w.p. } 1/2. \end{cases}$$

Suppose $\mathcal{F}_2 = (a, 2/d)$. Then A^* must align with a. There are $d/2$ choices of 1-sparse vectors which align with a, and therefore

$$\alpha_2(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_2) = \begin{cases} 0 & a' \text{ is not 1-sparse} \\ 2/d & a' \text{ is 1-sparse and aligns with } a \\ 0 & a' \text{ is 1-sparse and does not align with } a \end{cases}$$

The case when $\mathcal{F}_2 = (a, 0)$ is similar but with "aligns" and "does not align" interchanged. In either case we have

$$H(\alpha_2) = -\frac{d}{2} \cdot \frac{2}{d} \log\left(\frac{2}{d}\right) = \log d - \log 2.$$

Therefore

$$g_1(a) = \log d - (\log d - \log 2) = \log 2,$$

for $a \neq A^*$. Since $\pi_1^{\text{IDS}}(A^*) = 0$, we need not calculate $g_1(A^*)$.
Putting this together, for IDS we get

$$\Psi_1(\pi_1^{\text{IDS}}) = \frac{\Delta_1(\pi_1^{\text{IDS}})^2}{g_1(\pi_1^{\text{IDS}})} = \frac{(1 - 1/d)^2}{\log 2} \sim \frac{1}{\log 2}.$$

Example 0.4 – Assortment optimization

- Consider customers of unknown type $\theta \in \Theta$ with $|\Theta| = n$. Each product is aimed at customers of a given type, and the assortment $a \in \mathcal{A} = \Theta^m$ of m products offered is characterized by the vector $a = (a_1, \dots, a_m)$. Customers are more likely to derive high value from a product geared towards their type. When offered an assortment a, the customer associates with the ith product utility

$$U_{\theta,i}(a)^{(t)} = \beta \mathbb{1}_{a_i = \theta} + W_i^{(t)},$$

where $W_i^{(t)}$ follows an extreme-value distribution and $\beta \in \mathbb{R}$ is a known constant. The probability a customer of type θ chooses product i is

$$\frac{\exp(\beta \mathbb{1}_{a_i=\theta}}{\sum_{j=1}^m \exp(\beta \mathbb{1}_{a_j=\theta}}.$$

When an assortment a is offered at time t, the customer makes a choice $I_t = \text{argmax}_i\, U_{\theta,i}^{(t)}(a)$ and leaves review $U_{\theta,I_t}^{(t)}(a)$ indicating the utility they derive, both of which are observed by the recommendation system. The reward is the normalized utility, $U_{\theta,I_t}^{(t)}(a)/\beta$.
- If the type of customer θ were known, the optimal recommendation would be $A^* = (\theta, \theta, \dots, \theta)$, only products targeted at the customer's type. So both TS and UCB would only offer assortments consisting of a single type of product, and hence each require on the order of n samples to learn the customer's true type. IDS goes for a diverse assortment to learn more quickly.
- Suppose θ is drawn uniformly from Θ and consider $\beta \to \infty$. Then the probability a customer chooses a product of type θ (if it is available) tends to 1, and the normalised review $\beta^{-1} U_{\theta,I_t}^{(t)}(a)$ tends to $\mathbb{1}_{a_{I_t}=\theta}$, an indicator of whether the chosen product is of type θ. The initial assortment offered by IDS will be m different and previously untested product types, since this maximizes the algorithm's expected gain in the next period and the information gain. The algorithm continues offering assortments containing m unique, untested types

until a review near $U^{(t)}_{\theta,I_t}(a) \approx 1$ is received. With high probability this takes at most $\lceil n/m \rceil$ time periods — an factor of m faster than UCB or TS.

- Again we explicitly calculate the information ratio of each policy.
  - For Thompson sampling: we first find $\Delta_1(\pi_1^{\mathrm{TS}})^2$. We know

$$R_{1,A^*} = 1, \quad R_{1,a} = \begin{cases} 0 & a \neq A^* \\ 1 & a = A^* \end{cases} \implies \Delta_1(a) = \mathbb{1}_{a \neq A^*},$$

and

$$\pi_1^{\mathrm{TS}}(a) = \begin{cases} 1/n & a = (\theta, \theta, \ldots, \theta) \text{ for some } \theta \in \Theta \\ 0 & \text{otherwise,} \end{cases}$$

where we recall $|\Theta| = n$. Therefore

$$\Delta_1(\pi_1^{\mathrm{TS}}) = \sum_{\theta \in \Theta} \pi_1^{\mathrm{TS}}((\theta, \ldots, \theta)) \mathbb{1}_{a \neq A^*} = (n-1) \cdot \frac{1}{n} = 1 - \frac{1}{n}.$$

So $\Delta_1(\pi_1^{\mathrm{TS}})^2 = (1 - 1/n)^2$.
For $g_1(\pi_1^{\mathrm{TS}})$, first consider $g_1(a)$, $a \neq A^*$. We have

$$\begin{aligned} g_1(a) &= \mathbb{E}[H(\alpha_1) - H(\alpha_2) \mid \mathcal{F}_1,\, A_1 = a \neq A^*] \\ \alpha_1(a') &= \mathbb{P}(A^* = a') = \begin{cases} 1/n & \exists \theta \in \Theta : a' = (\theta, \ldots, \theta) \\ 0 & \text{otherwise} \end{cases} \\ \implies H(\alpha_1) &= \log n \\ \alpha_2(a') &= \mathbb{P}(A^* = a' \mid \mathcal{F}_2 = (a, 0)) = \begin{cases} 0 & a' = a \\ \frac{1}{n-1} & \exists \theta \in \Theta : a' = (\theta, \ldots, \theta),\, a' \neq a \\ 0 & \text{otherwise} \end{cases} \\ \implies H(\alpha_2) &= \log(n-1) \\ \implies g_1(a) &= \log\left(\frac{n}{n-1}\right),\ a \neq A^*. \end{aligned}$$

For $a = A^*$, we have

$$\begin{aligned} g_1(A^*) &= \mathbb{E}[H(\alpha_1) - H(\alpha_2) \mid \mathcal{F}_1,\, A_1 = A^*] \\ H(\alpha_1) &= \log n \\ \alpha_2(a') &= \mathbb{P}(A^* = a' \mid \mathcal{F}_2 = (a, 1)) = \begin{cases} 1 & a' = a \\ 0 & \text{otherwise} \end{cases} \\ \implies H(\alpha_2) &= 0 \\ \implies g_1(A^*) &= \log n. \end{aligned}$$

Putting this together we have

$$g_1(\pi_1^{\mathrm{TS}}) = \frac{1}{n}\log(n) + \frac{n-1}{n}\log\left(\frac{n}{n-1}\right),$$

so that

$$\Psi_1(\pi_1^{\mathrm{TS}}) = \frac{\left(1 - \frac{1}{n}\right)^2}{\frac{1}{n}\log(n) + \frac{n-1}{n}\log\left(\frac{n}{n-1}\right)} \sim \frac{n}{\log(n)}.$$

  - For IDS: we abuse notation slightly and write $\theta \in a$ to mean that one of the entries of $a \in \mathcal{A} = \Theta^m$ is the element $\theta \in \Theta$. To calculate $\Delta_1(\pi_1^{\mathrm{IDS}})$, note we have

$$R_{1,A^*} = 1, \quad R_{1,a} = \mathbb{1}_{\theta \in a}, \quad R_{1,A^*} - R_{1,a} = \mathbb{1}_{\theta \notin a}.$$

Moreover, there are $m! \cdot \binom{n}{m} = \frac{n!}{(n-m)!}$ different m-tuples from n symbols, so

$$\pi_1^{\text{IDS}}(a) = \begin{cases} \frac{(n-m)!}{n!} & \text{all elements of } a \text{ are different} \\ 0 & \text{otherwise,} \end{cases}$$

so

$$\begin{aligned} \Delta_1(\pi_1^{\text{IDS}}) &= \sum_a \pi_1^{\text{IDS}}(a) \mathbb{1}_{\theta \notin a} = \sum_{a\text{: all elements of } a \text{ differ}} \frac{(n-m)!}{n!} \mathbb{1}_{\theta \notin a} \\ &= \frac{(n-1)!}{(n-m-1!)} \cdot \frac{(n-m)!}{n!} = \frac{n-m}{n} = \left(1 - \frac{m}{n}\right), \end{aligned}$$

where to get to the second line we use the fact there are $m! \cdot \binom{n-1}{m} = \frac{(n-1)!}{(n-m-1)!}$ choices of m-tuples with different symbols which do not contain θ.

Now we compute $g_1(\pi_1^{\text{IDS}}) = \mathbb{E}[H(\alpha_1) - H(\alpha_2) \mid \mathcal{F}_1, A_1 = a]$. Again we note that $\mathcal{F}_1 = \varnothing$ so conditioning on it has no effect.

$$\alpha_1(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_1) = \begin{cases} 1/n & \exists \theta \in \Theta,\, a' = (\theta, \ldots, \theta) \\ 0 & \text{otherwise,} \end{cases}$$

so $H(\alpha_1) = \log n$. For α_2, there are two cases depending on a. If $\theta \in a$, then $\mathcal{F}_2 = (A_1, 1)$ and we have

$$\alpha_2(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_2) = \begin{cases} 1/m & \exists \theta \in A_1,\, a' = (\theta, \ldots, \theta) \\ 0 & \text{otherwise,} \end{cases}$$

so $H(\alpha_2) = \log m$. If instead $\theta \notin a$, then $\mathcal{F}_2 = (A_1, 0)$ and we have

$$\alpha_2(a') = \mathbb{P}(A^* = a' \mid \mathcal{F}_2) = \begin{cases} 1/(n-m) & \exists \theta \notin A_1,\, a' = (\theta, \ldots, \theta) \\ 0 & \text{otherwise,} \end{cases}$$

so $H(\alpha_2) = \log(n-m)$. Taken together, we get

$$g_1(a) = \begin{cases} \log(n/m) & \theta \in a \\ \log(n/(n-m)) & \theta \notin a. \end{cases}$$

Finally we compute

$$g_1(\pi_1^{\text{IDS}}) = \sum_a \pi_1^{\text{IDS}}(a) g_1(a).$$

To do so we recall that

$$\pi_1^{\text{IDS}}(a) = \begin{cases} \frac{(n-m)!}{n!} & \text{all elements of } a \text{ are different} \\ 0 & \text{otherwise,} \end{cases}$$

and so we must count the number of tuples in $\mathcal{A} = \Theta^m$ which have all different elements and which either contain θ or do not contain θ. For the number of tuples containing θ, there are $\binom{n-1}{m-1}$ choices for the other symbols and $m!$ ways to arrange them (noting they are all different), so there are $m!\binom{n-1}{m-1}$ such sequences. For the number of tuples not containing θ, there are $\binom{n-1}{m}$ choices for the symbols and $m!$ ways to arrange them, so there are $m!\binom{n-1}{m}$ such sequences. Consequently

$$\begin{aligned} g_1(\pi_1^{\text{IDS}}) &= m!\binom{n-1}{m-1} \frac{(n-m)!}{n!} \log\left(\frac{n}{m}\right) + m!\binom{n-1}{m} \frac{(n-m)!}{n!} \log\left(\frac{n}{n-m}\right) \\ &= \frac{m!(n-1)!}{(m-1)!(n-m)!} \cdot \frac{(n-m)!}{n!} \log\left(\frac{n}{m}\right) + \frac{m!(n-1)!}{m!(n-m-1)!} \cdot \frac{(n-m)!}{n!} \log\left(\frac{n}{n-m}\right) \\ &= \frac{m}{n} \log\left(\frac{n}{m}\right) + \frac{n-m}{n} \log\left(\frac{n}{n-m}\right), \end{aligned}$$

so that finally we have

$$\Psi_1(\pi_1^{\text{IDS}}) = \frac{\Delta_1(\pi_1^{\text{IDS}})^2}{g_1(\pi_1^{\text{IDS}})} = \frac{\left(1 - \frac{m}{n}\right)^2}{\frac{m}{n} \log\left(\frac{n}{m}\right) + \frac{n-m}{n} \log\left(\frac{n}{n-m}\right)} \leq \frac{n}{m \log\left(\frac{n}{m}\right)}.$$

Could also try to maximize information about model parameter θ, e.g. suppose we try to maximize

$$\mathbb{E}_t[R_{t,a}] + \lambda I_t(Y_{t,a};\theta);$$

call this θ-IDS. Next example shows it isn't good:

Example 0.5 – Unconstrained assortment optimization

- Again consider recommending assortments of products to a customer with unknown preferences. Can choose any subset $a \subset \{1,\dots,n\}$ to display. If offered assortment a at time t, customer chooses $J_t = \operatorname{argmax}_{i\in a}\theta_i$ and leaves review $R_{t,a} = \theta_{J_t}$ where θ_i is utility for product i. Recommendation system gets both J_t and review $R_{t,a}$, and wants to learn the assortment maximising the review $R_{t,a}$. Suppose θ is a random permutation of $(1, 1/2, 1/3, \dots, 1/n)$. Customer is known to assign utility 1 to best item, 1/2 to the next best, and so on, but the rank ordering is unknown.
- No learning needed to offer optimal assignment: just offer full collection $a = \{1,\dots,n\}$, customer picks most preferred. This assortment reveals the item most preferred to the customer but does not yield information about other items.
- θ-IDS starts by offering full assortment $A_1 = \{1,\dots,n\}$ yielding a reward of 1, and by revealing top item, yields information $I_1(Y_{1,A_1};\theta) = \log n$. If $1/2 < \lambda\log(n-1)$, which is guaranteed for large enough n, it will continue to experiment with suboptimal assortments; in the second time step, it offers A_2, consisting of all products except $\operatorname{argmax}_i\theta_i$. This reveals the second most preferred item, yielding information $I_2(Y_{2,A_2};\theta) = \log(n-1)$. This process repeats until the first k such that $\lambda\log(n+1-k) < 1 - 1/k$.
- IDS recognises that the optimal assortment $A^* = \{1,\dots,n\}$ does not depend on full knowledge of θ, so does not invest in identifying θ.

Expected improvement and knowledge gradient

- Expected improvement: one of the most used techniques in Bayesian optimisation. Let $\mu_{t,a} = \mathbb{E}[R_{t,a} \mid \mathcal{F}_t]$ to be the expected reward under the posterior, and $V_t = \max_{a'}\mu_{t,a'}$ to be best objective value attainable under current information. Expected improvement of action a is defined to be
$$\mathbb{E}[\max\{f_\theta(a), V_t\} \mid \mathcal{F}_t],$$
where $f_\theta(a) = \mathbb{E}[R_{t,a} \mid \theta]$ is the expected reward generated by action a under unknown true parameter θ. EGO algorithm aims to identify high-performing actions by sequentially sampling those that yield the highest expected improvement. Like UCB, this encourages selecting actions that could perform well. However, doesn't place value on indirect information.
- Knowledge gradient uses a modified improvement measure. At time t, it computes
$$v_{t,a}^{KG} := \mathbb{E}[V_{t+1} \mid \mathcal{F}_t, A_t = a] - V_t$$
for each action a. If V_t measures the quality of decision that can be made based on current information, then $v_{t,a}^{KG}$ captures immediate improvement in decision quality due to sampling action a and observing $Y_{t,a}$. For a problem with time horizon T, the knowledge gradient policy selects an action in time period t by maximizing $\mu_{t,a} + (T-t)v_{t,a}^{KG}$ over actions $a \in \mathcal{A}$. This measure $v_{t,a}^{KG}$ of the value of sampling an action places value on indirect information; even if an action is known to yield low expected reward, it could increase V_t by providing information about other actions.

Example 0.6 – A known standard

- Consider a problem with two actions $\mathcal{A} = \{a_1, a_2\}$, where rewards from a_1 are Ber(1/2) and rewards from a_2 are either Ber(3/4) with prior probability p_0 and Ber(1/4) with prior probability $1-p_0$.
- Action 1 has a mean reward of 1/2. When $p_0 \le 1/4$, upon sampling action 2 and observing a reward of 1, then posterior expected reward of action 2 becomes
$$\mathbb{E}[R_{2,a_2} \mid R_{1,a_2} = 1] = \frac{p_0(3/4)}{p_0(3/4) + (1-p_0)(1/4)} \le 1/2.$$
So a single sample is never enough to change which action has the highest posterior expected

reward, so $v^{KG}_{t,a_2} = 0$ and the KG decision rule selects action 1 in the first period. Since nothing is learned in this interaction, KG continues to select action 1 in all subsequent periods. Cumulative regret is therefore $(p_0/4)T$, growing linearly in T.
- Sampling action 2 will not immediately shift the decision-maker's prediction of the best action, but these samples influence their posterior beliefs and reduce uncertainty about which action is optimal. So IDS assigns positive probability to sampling the second action.

- Modifications to KG have been proposed (e.g. KG*) which aim to fix this, but may explore very inefficiently. Moreover, it depends how ties are broken among actions which maximize $\mu_{T,a} + (T-t)v^{KG}_{t,a}$.

Example 0.7 – Sparse linear model with an outside option

- Suppose $\mathcal{A} = \{O\} \cup \mathcal{A}'$, where O is some outside option known to yield reward 1/2 and $\mathcal{A}' = \left\{x/\|x\|_1 : x \in \{0,1\}^d,\, x \neq 0\right\}$. The reward generated by $a \in \mathcal{A}'$ is $a^\intercal\theta$ for some θ drawn uniformly from 1-sparse vectors $\Theta = \left\{\theta' \in \{0,1\}^d : \|\theta\|_0 = 1\right\}$. Assume $d = 2^m$ for some $m \in \mathbb{N}$.
- When the horizon T is long, the inclusion of the outside option should be irrelevant; nothing is learned by sampling the outside option and it is known apriori to be suboptimal. An optimal algorithm should sample actions in $\mathcal{A}'$ until the optimal action is found and should minimize the regret incurred. Without observation noise, KG and KG* are equivalent. We will see that when T is large, KG samples actions in $\mathcal{A}'$ until the optimal action is identified, but the expected number of samples and the expected regret both scale linearly with the dimension d. IDS identifies the optimal action after only $\log_2(d)$ steps.
- Note that unless the agent has exactly identified θ, selecting the outside option always generates the highest expected reward in the next period; only selecting an action with a single positive component could immediately reveal θ, so KG only places value on the information generated by such actions. When the horizon is large, KG prioritizes such information over the safe reward offered by the outside option, so engages in exhaustive search.
- IDS performs binary search. In the first period, it selects some permutation of $a = (2/d, \ldots, 2/d, 0, \ldots, 0)$. By observing either $a^\intercal\theta = 0$ or $a^\intercal\theta = 1$ it rules out half the parameter vectors in Θ_1. Continuing this search, the parameter is identified in $O(\log_2(d))$ steps. To see why, note that selecting an action with $d/2$ positive components offers strictly maximal information gain $I_1(A^*; a^\intercal\theta) = \log 2$ and weakly maximal reward $\mathbb{E}[a^\intercal\theta] = 1/d$ among all actions $a \in \mathcal{A}'$, so any action in $\mathcal{A}'$ not a permutation of a is strictly dominated and is never sampled by IDS. Also, IDS selects O with probability $1-\alpha^*$, where α^* is the minimizer of the information ratio

$$\alpha^* = \text{argmin}_{\alpha\in[0,1]} \frac{((1-\alpha)/2 + \alpha(1-1/d))^2}{\alpha\log_2(d)} = \text{argmin}_{\alpha\in[0,1]} \frac{1}{2\sqrt{\alpha}} + \sqrt{\alpha}\left(\frac{1}{2} - \frac{1}{d}\right) = 1.$$

This process continues inductively.

0.6 Regret bounds

These results follow from information-theoretic analysis of TS.

0.6.1 General bound

A general bound for any policy in terms of its information ratio and entropy of optimal action distribution.

Proposition 0.8
For any policy π and time $T \in \mathbb{N}$,

$$\mathbb{E}[\text{Regret}(T,\pi)] \leq \sqrt{\bar{\Psi}_T(\pi) H(\alpha_1) T}$$

where

$$\bar{\Psi}_T(\pi) = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_\pi[\Psi_t(\pi_t)]$$

is the average expected information ratio under π.

Corollary 0.9

Fix a deterministic $\lambda \in \mathbb{R}$ and policy π such that $\Psi_t(\pi_t) \le \lambda$ almost surely for each $t \in \{1, \ldots, T\}$. Then

$$\mathbb{E}[\text{Regret}(T,\pi)] \le \sqrt{\lambda H(\alpha_1) T}.$$

0.6.2 Specialized bounds on minimal information ratio

To simplify exposition, assume rewards are uniformly bounded.

Worst case bound

Proposition 0.10

For any $t \in \mathbb{N}$, $\Psi_t(\pi_t^{\text{IDS}}) \le |\mathcal{A}|/2$ almost surely.

Taking section 0.6.2

Full information

Proposition 0.11

Suppose for each $t \in \mathbb{N}$ there is a random variable $Z_t : \Omega \to \mathbb{Z}$ such that for each $a \in \mathcal{A}$, $Y_{t,a} = (a, Z_t)$. Then for all $t \in \mathbb{N}$, $\Psi_t(\pi_t^{\text{IDS}}) \le 1/2$ almost surely.

0.7 Extensions

- IDS for discounted problems

Provably efficient RL with Rich Observations via Latent State Decoding

Du, Krishnamurthy, Jiang, Agarwal, Dudik, Langford [1]

0.8 Abstract

- Exploration problem in episodic MDPs, rich observations from a small number of latent states
- Estimate mapping from observations to latent states inductively via regression and clustering and use this for good exploration policies
- Finite-sample guarantees on quality of learned state decoding function and exploration policies
- Improves over Q-learning with naïve exploration

0.9 Introduction

- Study RL in episodic environments with rich observations (e.g. images, text); not many methods to explore well in these environments
- Previous approaches hard to adapt to rich observation spaces since number of interactions is polynomial in number of observed states
- Recent work on contextual decision processes identified low-rank structures enabling exploration algorithms with sample complexity polynomial in rank parameter; however provably computationally intractable or practically cumbersome
- Here: learn a decoding function mapping a rich observation to corresponding latent state. If such a function learned perfectly, reduce problem to a tabular problem where exploration is tractable. Show such an approach is
  - Provably sample-efficient: use number of samples polynomial in number of latent states, horizon, and complexity of decoding function class
  - Computationally practical: easy to implement
- Introduce block MDPs and ε-policy covers

0.10 Setting and task definition

- Write $[h] = \{1, \ldots, h\}$
- For finite S write $U(S)$ for the uniform distribution over S
- Write Δ_d for simplex in $\mathbb{R}^d$
- Write $\|\cdot\|$ and $\|\cdot\|_1$ for Euclidean and ℓ_1 norms

0.10.1 Block MDPs

Definition 0.12 – Block MDP

A *block MDP* is a finite (but unobservable) latent space $\mathcal{S}$, a finite action space $\mathcal{A}$ with $|A| = K$, and a possibly infinite observable context space $\mathcal{X}$. The dynamics of a BMDP are described by the initial state $s_1 \in \mathcal{S}$ and two conditional probability functions: the state-transition function $p(s' \mid s, a)$ and the context-emission function $q(x \mid s)$ for all $s, s' \in \mathcal{S}$, $a \in \mathcal{A}$, $x \in \mathcal{X}$. For continuous context spaces, $q(\cdot \mid s)$ is a density relative to some suitable measure.

We assume that each context x uniquely determines its generating state s, i.e. the context space $\mathcal{X}$ can be partitioned into disjoint blocks $\mathcal{X}_s$ each containing the support of the conditional distribution $q(\cdot \mid s)$.

- May also consider distributions of rewards conditioned on context and action, but this paper focuses on pure exploration.
- Consider episodic learning tasks, finite horizon H.

- Each step the environment generates a context $x_h \sim q(\cdot \mid s_h)$, agent observes context x_h (not s_h), takes action a_h, and environment transitions to a new state $s_{h+1} \sim p(\cdot \mid s_h, a_h)$. Call the sequence $(s_1, x_1, a_1, \ldots, s_H, x_H, a_H, s_{H+1}, x_{H+1})$ a trajectory.
- The sets $\mathcal{X}_s$ are unique up to sets of measure zero under $q(\cdot \mid s)$.
- Block structure implies there is a perfect decoding function $f^* : \mathcal{X} \to \mathcal{S}$ mapping contexts to generating states.
- BMDP model has been assumed in prior works but not under the same name.
- Assume that $\mathcal{S}$ can be partitioned into disjoint sets $\mathcal{S}_h$, $h \in [H+1]$, so that $p(\cdot \mid s, a)$ is supported on $\mathcal{S}_{h+1}$ whenever $s \in \mathcal{S}_h$. Refer to h as the level and assume it is observable as part of context, so context space is also partitioned into sets $\mathcal{X}_h$. Write $S_{[h]} = \cup_{l \in [h]} \mathcal{S}_l$ for set of states up to level h, similarly for $X_{[h]}$.
- Assume $|\mathcal{S}_h| \leq M$. Seek learning algorithms scaling polynomially in M, K and H, but not depending on $|\mathcal{X}|$ explicitly.

0.10.2 Solution concept: cover of exploratory policies

- For each state $s \in \mathcal{S}$, seek an agent strategy for reaching that state s. Here an agent strategy is an h-step policy, i.e. a map $\pi : \mathcal{X}_{[h]} \to \mathcal{A}$ specifying what action to take in each context up to step h. If $h < H$, the agent can act arbitrarily until the end of the episode.
- For h-step policy π, write $\mathbb{P}^\pi$ to denote distribution over h-step trajectories induced by π, e.g. $\mathbb{P}^\pi(s)$ is probability of reaching state s while executing π.
- Consider randomized strategies; an h-step policy mixture η is a distribution over h-step policies. To execute η, draw a policy $\pi \sim \eta$ and follow π. Induced distribution over h-step trajectories is denoted $\mathbb{P}^\eta$.
- Need to concatenate policies; write $\pi \odot a$ for the policy executing π and then choosing action a afterwards. Similarly, if η is a policy mixture and ν a distribution over $\mathcal{A}$, write $\eta \odot \nu$ for policy mixture equivalent to following a policy from η and then sampling and following an action from ν.

Definition 0.13 – Maximum reaching probability
For any $s \in \mathcal{S}$, its *maximum reaching probability* $\mu(s)$ is

$$\mu(s) := \max_{\pi} \mathbb{P}^\pi(s),$$

where the maximum is taken over all map $\mathcal{X}_{[H]} \to \mathcal{A}$. The policy attaining the maximum for a given s is denoted π_s^*.

- WLOG assume all states are reachable, i.e. $\mu(s) > 0$ for all s. Write $\mu_{\min} = \min_{s \in \mathcal{S}} \mu(s)$ for the $\mu(s)$-value of the hardest-to-reach-state. $\mathcal{S}$ is finite and all states are reachable so $\mu_{\min} > 0$.

Definition 0.14 – Policy cover of state space
Say a set of policies Π_h is an *ε-policy cover* of $\mathcal{S}_h$ if for all $s \in \mathcal{S}_h$ there is an $(h-1)$-step policy $\pi \in \Pi_h$ such that $\mathbb{P}^\pi(s) \geq \mu(s) - \varepsilon$. A set of policies Π is an *ε-policy cover* of $\mathcal{S}$ if it is an ε-policy cover of $\mathcal{S}_h$ for all $h \in [H+1]$.

- Seek policy cover of a small size (e.g. $O(|S|)$) and with small ε. Then we can reach every state with the largest possible probability (up to ε) by executing each policy in the cover in turn.

0.11 Embedding approach

- Key challenge: lack of access to latent s. Here: learn a decoding function f mapping contexts to latents.
- Still a hard unsupervised learning problem unless we make strong assumptions about $\mathcal{X}_s$ or the emission distributions $q(\cdot \mid s)$.
- Instead: make separability assumptions about latent transition probabilities p.

0.11.1 Embeddings and function approximation

- To construct f, learn low-dimensional representations of contexts and latents in a shared space Δ_{MK}. Learn embeddings $g : \mathcal{X} \to \Delta_{MK}$ for contexts and $\phi : \mathcal{S} \to \Delta_{MK}$ for states, hoping that $g(x)$ and $\phi(s)$ should be close iff $x \in \mathcal{X}_s$.
- Can construct g and ϕ via an essentially supervised approach. State embedding ϕ is a low complexity object whereas g has a high complexity, so limit g to some class $\mathcal{G} \subseteq \{\mathcal{X} \to \Delta_{MK}\}$, e.g. generalised linear models, tree ensembles, neural nets etc.
- Allowing a separate $g_h \in \mathcal{G}$ for each level we assume realizability:

  Definition 0.15 – Realizability
  For any $h \in [H+1]$ and $\phi : \mathcal{S}_h \to \Delta_{MK}$, there is $g_h \in \mathcal{G}$ such that $g_h(x) = \phi(s)$ for all $x \in \mathcal{X}_s$ and $s \in \mathcal{S}_h$.

  That is, $\mathcal{G}$ must be able to match any state-embedding function ϕ across all blocks $\mathcal{X}_s$.
- Therefore consider classes $\mathcal{G}$ which are a composition $\phi' \circ f$ where f is a decoding function from some class $\mathcal{F} \subseteq \{\mathcal{X} \to \mathcal{S}\}$ and ϕ' is any mapping $\mathcal{S} \to \Delta_{MK}$. Here f decodes context x to a state $f(x)$ which is then embedded by ϕ'. The realizability assumption works if $\mathcal{F}$ contains a perfect decoding function f^* s.t. $f^*(x) = s$ whenever $x \in \mathcal{X}_s$.
- Given $\mathcal{G}$, want to find a suitable context-embedding function in $\mathcal{G}$ using a number of trajectories proportional to $\log |\mathcal{G}|$ when $\mathcal{G}$ is finite, or some general notion of complexity when $\mathcal{G}$ is infinite. Here: assume $\mathcal{G}$ finite for ease.
- Only need ability to solve least-squares problems, so assume we have access to an algorithm for solving vector-valued least-squares regression over $\mathcal{G}$:

Definition 0.16 – ERM oracle
Let $\mathcal{G}$ be a function class mapping $\mathcal{X}$ to Δ_{MK}. An *empirical risk minimization oracle* for $\mathcal{G}$ is any algorithm that takes as input a data set $D = \{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathcal{X}$, $y_i \in \Delta_{MK}$, and computes

$$\operatorname{argmin}_{g \in \mathcal{G}} \sum_{(x,y) \in D} \|g(x) - y\|^2 .$$

0.11.2 Backward probability vectors and separability

Definition 0.17 – Backward probabilities
Given a distribution ν over (s_{h-1}, a_{h-1}), and for any $s \in \mathcal{S}_{h-1}$, $a \in \mathcal{A}$, and $s' \in \mathcal{S}_h$, the *backward probability* is

$$b_\nu(s, a \mid s') = \frac{p(s' \mid s, a)\nu(s, a)}{\sum_{\tilde{s},\tilde{a}} p(s' \mid \tilde{s}, \tilde{a})\nu(\tilde{s}, \tilde{a})}.$$

For given $s' \in \mathcal{S}_h$, we collect the probabilities $b_\nu(s, a \mid s')$ across all $s \in \mathcal{S}_{h-1}$, $a \in \mathcal{A}$ into the *backward probability vector* $b_v(s') \in \Delta_{MK}$, padding with zeroes if $|\mathcal{S}_{h-1}| < M$.

- At the core of the approach since they correspond to state embeddings $\phi(s)$ approximated by algorithms
- Require that $b_v(s')$ for different states $s' \in \mathcal{S}_h$ be sufficiently separated from one another:

Definition 0.18 – γ-separability
There exists $\gamma > 0$ such that for any $h \in \{2, \ldots, H+1\}$ and any distinct $s', s'' \in \mathcal{S}_h$, the backward probability vectors wrt the uniform distribution are separated by a margin of at least γ, i.e.

$$\left\| b_v(s') - b_v(s'') \right\|_1 \geq \gamma,$$

where $\nu = U(\mathcal{S}_{h-1} \times \mathcal{A})$.

- Can be shown this assumption holds with $\gamma = 2$ when latent-state transitions are deterministic, but class of γ-separable models is substantially larger
- Why $b_v(s')$ useful? They arise as solutions to a specific least-squares problem wrt data generated by a policy whose marginal distribution over (s_{h-1}, a_{h-1}) matches ν.

Theorem 0.19
Let $e_{(s,a)}$ denote the vector of the standard basis in $\mathbb{R}^{MK}$ corresponding to $(s,a) \in \mathcal{S}_{h-1} \times \mathcal{A}$. Let ν be a distribution supported on $\mathcal{S}_{h-1} \times \mathcal{A}$ and $\tilde{\nu}$ be a distribution over (s,a,x') defined by sampling $(s,a) \sim \nu$, $s' \sim p(\cdot \mid s,a)$ and $x' \sim q(\cdot \mid s')$. Let

$$g_h \in \operatorname{argmin}_{g \in \mathcal{G}} \mathbb{E}_{\tilde{\nu}} \left[\left\| g(x') - e_{(s,a)} \right\|^2 \right].$$

Then, assuming realizability, every minimizer g_h satisfies $g_h(x') = b_v(s')$ for all $x' \in \mathcal{X}_{s'}$ and $s' \in \mathcal{S}_h$.

- $\tilde{\nu}$ is the marginal distribution induced by a policy whose marginal over (s_{h-1}, a_{h-1}) matches ν. Any minimizer g_h yields context embeddings corresponding to state embeddings $\phi(s') = b_v(s')$.

0.12 Algorithm for separable BMDPs

- Algo proceeds inductively level by level. For each level h, we learn
  - the set of discovered latents $\hat{S}_h \subseteq [M]$ and a decoding function $\hat{f}_h : \mathcal{X} \to \hat{\mathcal{S}}_h$, letting us identify latent states at level h from observed contexts;
  - the estimated transition probabilities $\hat{p}(\hat{s}_h \mid \hat{s}_{h-1}, a)$ across all $\hat{s}_{h-1} \in \hat{\mathcal{S}}_{h-1}$, $a \in \mathcal{A}$, $\hat{s}_h \in \hat{S}_h$;
  - a set of $(h-1)$-step policies $\Pi_h = \{\pi_{\hat{s}}\}_{\hat{s} \in \hat{S}_h}$

Theorem 0.20
There is a bijection $\alpha_h : \hat{\mathcal{S}}_h \to \mathcal{S}_h$ such that the following conditions are satisfied for all $\hat{s} \in \hat{S}_{h-1}$, $a \in \mathcal{A}$, $\hat{s}' \in \hat{S}_h$, and $s = \alpha_{h-1}(\hat{s}_{h-1})$, $s' = \alpha_h(\hat{s}')$, where α_{h-1} is the bijection for the previous level:

- Accuracy of $\hat{f}_h$: $\mathbb{P}_{x' \sim q(\cdot \mid s')} \left[\hat{f}_h(x') = \hat{s}' \right] \geq 1 - \varepsilon_f$;
- Accuracy of $\hat{p}$:
$$\sum_{\hat{s}'' \in \hat{S}_h,\, s'' = \alpha_h(\hat{s}'')} \left| \hat{p}(\hat{s}'' \mid \hat{s}, a) - p(s'' \mid s, a) \right| \leq \varepsilon_p;$$
- Coverage by Π_h: $\mathbb{P}^{\pi_{\hat{s}'}}(s') \geq \mu(s') - \varepsilon$.

PCID constructs these level by level. Given these objects up to level $h-1$, next construction proceeds via:

1. Regression step: learn $\hat{g}_h$. Collect a dataset of trajectories by repeatedly following a policy mixture η_h. Use $\hat{f}_{h-1}$ to identify $\hat{s}_{h-1} = \hat{f}_{h-1}(x_{h-1})$ on each trajectory, getting samples $(\hat{s}_{h-1}, a_{h-1}, x_h)$ from $\tilde{\nu}$ induced by η_h. Context embedding $\hat{g}_h$ can then be obtained by solving the empirical version of
$$\min_{g \in \mathcal{G}} \mathbb{E}_{\tilde{\nu}} \left[\left\| g(x') - e_{(s,a)} \right\|^2 \right].$$
Choice of η_h ensures each state s_{h-1} is reached with probability at least $(\mu_{\min} - \varepsilon)/M$, which is bounded away from zero for ε sufficiently small.
2. Clustering step: learn $\hat{\phi}$ and $\hat{f}_h$. We expect that $\hat{g}_h(x') \approx g_h(x') = b_\nu(s')$ for the distribution $\nu(\hat{s}_{h-1}, a_{h-1})$ induced by η_h. So all contexts x' from same state s' have embedding vectors $\hat{g}_h(x')$ close to each other and to $b_\nu(s')$. By separability, we can therefore use clustering to identify all contexts generated by the same latent state, and this procedure is sample-efficient since the embeddings are low-dimensional vectors. Each cluster corresponds to some latent state s' and any vector $\hat{g}_h(x')$ from that cluster can be used to define the state embedding $\hat{\phi}(s')$. The decoding function $\hat{f}_h$ is defined to map any context x' to the state s' whose embedding $\hat{\phi}(s')$ is the closest to $\hat{g}_h(x')$.
3. Dynamic programming: construct Π_h. Finally, since we can identify states at level h via $\hat{f}_h$, we can use collected trajectories to learn an approximate transition model $\hat{p}(\hat{s}' \mid \hat{s}, a)$ up to level h. Can therefore use DP to find policies that approximately optimise the probability of reaching any specific state $s' \in \mathcal{S}_h$. DP finds policies $\psi_{\hat{s}'}$ that act by directly observing decoded latent states. Policies $\pi_{\hat{s}'}$ are obtained by composing $\psi_{\hat{s}'}$ with decoding functions $\left\{ \hat{f}_l \right\}_{l \in [h-1]}$.

Theorem 0.21 – Sample complexity
Fix any $\varepsilon = O\left(\frac{\mu_{\min}^3 \gamma}{M^4 K^3 H}\right)$ and failure probability $\delta > 0$. Set

$$\begin{aligned} N_g &= \tilde{\Omega}\left(\frac{M^4 K^4 H \log |\mathcal{G}|}{\varepsilon \mu_{\min}^3 \gamma^2}\right) \\ N_\phi &= \tilde{\Theta}\left(\frac{MK}{\mu_{\min}}\right) \\ N_p &= \tilde{\Omega}\left(\frac{M^2 K H^2}{\mu_{\min} \varepsilon^2}\right) \\ \tau &= \frac{\gamma}{30MK}. \end{aligned}$$

Then with probability at least $1 - \delta$, PCID returns an ε-policy cover of $\mathcal{S}$ with size at most MH.

Note the notation here suppresses factors polynomial in $\log M$, $\log K$, $\log H$ and $\log(1/\delta)$.

Bibliography

[1] Simon S. Du et al. “Provably efficient RL with Rich Observations via Latent State Decoding”. In: *CoRR* abs/1901.09018 (2019). arXiv: 1901.09018. URL: http://arxiv.org/abs/1901.09018 (cited on page 12).

[2] Philipp Hennig and Christian J. Schuler. *Entropy Search for Information-Efficient Global Optimization.* 2011. arXiv: 1112.1217 [stat.ML]. URL: https://arxiv.org/abs/1112.1217 (cited on page 1).

[3] Daniel Russo and Benjamin Van Roy. *Learning to Optimize via Information-Directed Sampling.* 2017. arXiv: 1403.5556 [cs.LG]. URL: https://arxiv.org/abs/1403.5556 (cited on page 1).

[4] Julien Villemonteix, Emmanuel Vazquez, and Eric Walter. *An informational approach to the global optimization of expensive-to-evaluate functions.* 2007. arXiv: cs/0611143 [cs.NA]. URL: https://arxiv.org/abs/cs/0611143 (cited on page 1).