A New Theoretical Framework for Fast and Accurate Online Decision-Making

Anonymous Author(s)
Affiliation
Address
email

Abstract

We introduce a novel theoretical framework for Return On Investment (ROI) maximization in repeated decision-making. Our setting is motivated by the use case of companies that regularly receive proposals for technological innovations and want to quickly decide whether they are worth implementing. We design an algorithm for learning ROI-maximizing decision-making policies over a sequence of innovation proposals. Our algorithm provably converges to an optimal policy in class Π at a rate of order $\min\left\{1/(N\Delta^2), N^{-1/3}\right\}$, where N is the number of innovations and Δ is the suboptimality gap in Π. A significant hurdle of our formulation, which sets it aside from other online learning problems such as bandits, is that running a policy does not provide an unbiased estimate of its performance.

1 Introduction

Often, companies have to make yes/no decisions, such as whether to adopt a new technology or retire an old product. However, finding out the best option in all circumstances could mean spending too much time or money in the evaluation process. If the decisions to make are many, one could be better off making more of them quickly and inexpensively, provided that these decisions have an overall positive effect. In this paper, we investigate the problem of determining a decision policy to balance the reward over cost ratio optimally (i.e., to maximize the return on investment).

A motivating example. Consider a technology company that keeps testing innovations to increase some chosen metric (e.g., benefits, gross revenue, revenue excluding the traffic acquisition cost). Before deploying an innovation, the company wants to figure out whether it is profitable. As long as each innovation can be tested on i.i.d. samples of users, the company can perform randomized tests and make statistically sound decisions. However, there is an incentive to make these tests run as quickly as possible because, for example, the testing process is expensive. Another reason could be that keeping a team on a project that has negative, neutral, or even borderline positive potential prevents it from testing other ideas that might lead to a significantly better improvement. In other words, it is crucial to learn when to drop barely positive innovations in favor of highly positive ones, so to increase the overall flow of improvement over time (i.e., the ROI of the tests).

More generally, our framework describes problems where an agent faces a sequence of decision tasks consisting of either accepting or rejecting an innovation. Before making each decision, the agent can invest resources into reducing the uncertainty on the value brought by the innovation. The global objective is to maximize the total ROI. Namely, the ratio between the total value accumulated by accepting innovations and the total cost. For an in-depth discussion on alternative goals, we refer the reader to the Supplementary Material (Section E).

The model. Each task n in the sequence is associated with a pair $(\mu_n, \mathcal{D}_n)$ that the learner can *never* directly observe.

- μ_n is a random variable representing the (possibly negative) true value of the n-th innovation.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

- $\mathcal{D}_n$ is a probability distribution over the real numbers with expectation μ_n, modeling the feedback on the n-th innovation that the learner can gather from testing (see below).

During the n-th task, the learner can draw arbitrarily many i.i.d. samples $X_{n,1}, X_{n,2}, \ldots$ from $\mathcal{D}_n$, accumulating information on the unknown value μ_n of the innovation currently being tested. After stopping drawing samples, the learner can decide to either accept the innovation, earning μ_n as a reward, or reject it and gain nothing instead. We measure the agent performance during N tasks as the (expected) total amount of value accumulated by accepting innovations μ_n divided by the (expected) total number of samples requested throughout all tasks. In Section 3 we present this setting in more detail and introduce the relevant notation.

I.I.D. assumption. We assume that the value μ_n of the n-th innovation is drawn i.i.d. from an unknown and fixed distribution. This assumption is meaningful if past decisions do not influence future innovations whose global quality remains stable over time. In particular, it applies whenever innovations can progress in many orthogonal directions, each yielding a similar added value (e.g., when different teams within the same company test improvements relative to individual aspects of the company). If both the state of the agent and that of the environment evolve, but the ratio of good versus bad innovations remains essentially the same, then this i.i.d. assumption is still justified. In other words, it is not necessarily the absolute quality of the innovations that has to remain stationary, but rather the relative added value of the innovations given the current state of the system. This case is frequent in practice, especially when a system is close to its technological limit. Last but not least, algorithms designed under stochastic assumptions often perform surprisingly well in practice, even if i.i.d. assumptions are not fully satisfied or simply hard to check.

A baseline strategy and policy classes. A natural, yet suboptimal, approach for deciding if an innovation is worth accepting is to gather samples sequentially, stopping as soon as the absolute value of their running average surpasses a threshold, and then accepting the innovation if and only if the average is positive. The major drawback of this approach is that the value μ_n of an innovation n could be arbitrarily close to zero. In this case, the number of samples needed to reliably determine its sign (which is of order $1/\mu_n^2$) would become prohibitively large. This would result is a massive time investment for an innovation whose return is negligible at best. In hindsight, it would have been better to reject the innovation early and move on to the next task. For this reason, testing processes in practice needs hard termination rules of the form: *if after drawing a certain number of samples no confident decision can be taken, then terminate the testing process rejecting the current innovation.* Denote by τ this capped early stopping rule and by accept the accept/reject decision rule that comes with it. We say that the pair $\pi = (\tau, \mathrm{accept})$ is a *policy* because it fully characterize the decision-making process for an innovation. Policies defined by capped early stopping rules (see (4) for a concrete example) are of great practical importance (Johari et al., 2017; Kohavi et al., 2013). However, policies can be defined more generally by any reasonable stopping rule and decision function. Given a (possibly infinite) set of policies, and assuming that $\mu_1, \mu_2, \ldots$ are drawn i.i.d. from some unknown but fixed distribution, the goal is to learn efficiently, at the lowest cost, the best policy $\pi_\star$ in the set with respect to a sensible metric. Competing against fixed policy classes is a common modeling choice that allows to express the intrinsic constraints that are imposed by the nature of the decision-making problem. For example, even if some policies outside of the class could theoretically yield better performance, they might not be implementable because of time, budget, fairness, or technology constraints.

Challenges. One of the biggest challenges arising in our framework is that running a decision-making policy generates a collection of samples that —in general— cannot be used to form an unbiased estimate of the policy reward (see the impossibility result in Section F of the Supplementary Material). The presence of this bias is a significant departure from settings like multiarmed and firing bandits (Auer et al., 2002; Jain and Jamieson, 2018), where the learner observes an unbiased sample of the target quantity at the end of every round (see the next section for additional details). Moreover, contrary to standard online learning problems, the relevant performance measure is neither additive in the number of innovations nor in the number of samples per innovation. Therefore, algorithms have to be analyzed globally, and bandit-like techniques —in which the regret is additive over rounds— cannot be directly applied. We argue that these technical difficulties are a worthy price to pay in order to define a plausible setting, applicable to real-life scenarios.

Main contributions. The first contribution of this paper is providing a mathematical formalization of our ROI maximization setting for repeated decision making (Section 3). We then design an

algorithm called Capped Policy Elimination (Algorithm 1, CAPE) that applies to finite policy classes (Section 4). We prove that CAPE converges to the optimal policy at rate $1/(\Delta^2 N)$, where N is the number of tasks and Δ is the unknown gap between the performance of the two best policies, and at rate $N^{-1/3}$ when Δ is small (Theorem 1) . In Section 5 we tackle the challenging problem of infinitely large policy classes. For this setting, we design a preprocessing step (Algorithm 2, ESC) that leads to the ESC-CAPE algorithm. We prove that this algorithm converges to the optimal policy in an infinite set at a rate of $N^{-1/3}$ (Theorem 4).

Limitations. Although we do not investigate lower bounds in this paper, we conjecture that our $N^{-1/3}$ convergence rate it is optimal due to similarities with bandits with weakly observable feedback graphs (see Section 4, "Divided we fall"). Another limitation of our theory is that it only applies to i.i.d. sequences of values μ_n. It would be interesting to extend our analysis to distributions of μ_n that evolve over time. These two intriguing problems are left open for future research.

2 Related Work

Return on Investment (ROI) was developed and popularized by Donaldson Brown in the early Nineties (Flesher and Previts, 2013) and it is still considered an extremely valuable metric by the overwhelming majority of marketing managers (Farris et al., 2010). Beyond economics, mathematics, and computer science, ROI finds applications in other fields, such as cognitive science and psychology (Chabris et al., 2009). Despite this, to the best of our knowledge, no theoretical online learning framework has been developed specifically for ROI maximization. However, our novel formalization of this sequential decision problem does share some similarities with other known online learning settings. In this section, we review the relevant literature regarding these settings and stress the differences with ours.

Prophet inequalities and Pandora's box. In prophet inequalities (Lucier, 2017; Correa et al., 2019; Alaei et al., 2012), an agent observes sequentially (usually non-negative) random variables $Z_1, \dots, Z_n$ and decides to stop at some time τ; the reward is then Z_τ. Variants include the possibility of choosing more than one random variable (in which case the reward is some function of the selected random variables), and the possibility to go back in time (to some extent). The Pandora's box problem is slightly different (Weitzman, 1979; Kleinberg et al., 2016; Esfandiari et al., 2019); in its original formulation, the agent can pay a cost $c_n \geq 0$ to observe any Z_n. After stopping exploring, the agent's final utility is the maximum of the observed Z_n's minus the cumulative cost (or, in other variants, some function of these). Similarly to the (general) prophet inequality, the agent in our sequential problem faces random variables ($Z_n = \mu_n$ in our notation) and sequentially selects any number of them (possibly with negative values) without the possibility to go back in time and change past decisions. The significant difference is that the agent in our setting never observes the value of μ_n. In Pandora's box, the agent can see this value by paying some price (that approximately scales as $1/\varepsilon^2$ where ε is the required precision). Finally, the global reward is the cumulative sum (as in prophets) and not the maximum (as in Pandora's box) of the selected variables, normalized by the total cost (as in Pandora's box, but our normalization is multiplicative instead of additive, as it represents a ROI).

Multi-armed bandits. If we think of the set of all policies used by the agent to determine whether or not to accept innovations as arms, our setting becomes somewhat reminiscent of multi-armed bandits (Slivkins, 2019; Bubeck and Cesa-Bianchi, 2012; Rosenberg et al., 2007). However, there are several notable differences between these two problems. In stochastic bandits, the agent observes an unbiased estimate of the expected reward of each pulled arm. In our setting, the agent not only does not see it directly, but it is mathematically impossible to define such an estimator solely with the feedback received (see the impossibility result in Section F of the Supplementary Material). Hence, off-the-shelf bandit algorithms cannot be run to solve our problem. In addition, the objective in bandits is to maximize the cumulative reward, which is additive over time, while the ROI is not. Thus, it is unclear how formal guarantees for bandit algorithms would translate to our problem.

We could also see firing bandits (Jain and Jamieson, 2018) as a variant of our problem, where μ_n belongs to $[0, 1]$, $\mathcal{D}_n$ are Bernoulli distribution with parameter μ_n, and policies have a specific form that allows to easily define unbiased estimates of their rewards (which, we reiterate, is not possible in our setting in general). Furthermore, in firing bandits, it is possible to go back and forth in time, sampling from any of the past distributions $\mathcal{D}_n$ and gathering any number of samples from it. This is a reasonable assumption for the original motivations of firing bandits because the authors thought of μ_n as the value of a project in a crowdfunding platform, and, in their setting, drawing samples

from $\mathcal{D}_n$ corresponds to displaying projects on web pages. However, in our setting, μ_n represents the theoretical increment (or decrement) of a company's profit through a given innovation, and it is unlikely that a company would show new interest in investing in a technology that has been tested before and did not prove to be useful (a killed project is seldom re-launched). Hence, when the sampling of $\mathcal{D}_n$ stops, an irrevocable decision is made. After that, the learner cannot draw any more samples in the future. Finally, as in multi-armed bandits, the performance criterion in firing bandits is the cumulative reward and not the global ROI.

Another online problem that shares some similarities with ours is bandits with knapsacks (Badanidiyuru et al., 2018). In this problem, playing an arm consumes one unit of time together with some other resources, and the learner receives an unbiased estimate of its reward as feedback. The process ends as soon as time or any one of the other resources is exhausted. As usual, the goal is to maximize the cumulative regret. As it turns out, we can also think of our problem as a budgeted problem. In this restatement, there is a budget of T samples. The repeated decision-making process proceeds as before, but it stops as soon as the learner has drawn a total of T samples across all decision tasks. The goal is again to maximize the total expected reward of accepted innovations divided by T (see Section E of the Supplementary Material for more details on the reduction). As per the other bandit problems, there are two crucial differences. First, running a policy does not reveal an unbiased estimate of its reward. Second, our objective is different, and regret bounds do not directly imply convergence to optimal ROI.

Repeated A/B testing. We can view our problem as a framework for repeated A/B testing (Tukey, 1953; Genovese et al., 2006; Foster and Stine, 2008; Heesen and Janssen, 2016; Javanmard et al., 2018; Azevedo et al., 2018; Li and Barber, 2019; Schmit et al., 2019), in which assessing the value of an innovation corresponds to performing an A/B test, and the goal is maximizing the ROI. A popular metric to optimize sequential A/B tests is the so-called *false discovery rate* (FDR) —see (Ramdas et al., 2017; Yang et al., 2017) and references therein. Roughly speaking, the FDR is the ratio of accepted μ_n that are negative over the total number of accepted μ_n (or more generally, the number of incorrectly accepted tests over the total number if the metric used at each test changes with time). This, unfortunately, disregards the relative values of tests μ_n that must be taken into account when optimizing a single metric (Chen and Kasiviswanathan, 2020; Robertson and Wason, 2018). Indeed, the effect of many even slightly negative accepted tests could be overcome by a few largely positive ones. For instance, assume that the samples $X_{n,i}$ of any distribution $\mathcal{D}_n$ belong to $\{-1, 1\}$, and that their expected value μ_n is uniformly distributed on $\{-\varepsilon, \varepsilon\}$. To control the FDR, each A/B test should be run for approximately $1/\varepsilon^2$ times, yielding a ratio of the average value of an accepted test to the number of samples of order ε^3. A better strategy, using just one sample from each A/B test, is simply to accept μ_n if and only if the first sample is positive. Direct computations show that this policy, which fits our setting, achieves a significantly better performance of order ε.

Some other A/B testing settings are more closely related to ours, but make stronger additional assumptions or suppose preliminary knowledge: for example, smoothness assumptions can be made on both $\mathcal{D}_n$ and the distributions of μ_n (Azevedo et al., 2018), or the distribution of μ_n is known, and the distribution of samples belongs to a single parameter exponential family, also known beforehand (Schmit et al., 2019).

Rational metareasoning. Our setting is loosely related to the AI field of meta-reasoning (Griffiths et al., 2019; Hay et al., 2012). In a metalevel decision problem, determining the utility (or reward) of a given action is computationally intractable. Instead, the learner can run a simulation, investing a computational cost to gather information about this hidden value. The high-level idea is then to learn *which* actions to simulate. After running some simulations, the learner picks an action to play, gains the corresponding (hidden) reward, and the state of the system changes. In rational meta-reasoning, the performance measure is the value of computation (VOC): the difference between the increment in expected utility gained by executing a simulation and the cost incurred by doing so. This setting is not directly comparable to ours for two reasons. First, the performance measure is different, and the additive nature of the difference that defines the VOC gives no guarantees on our multiplicative notion of ROI. Second, in this problem, one can pick which actions to simulate, while in our settings, innovations come independently of the learner, who has to evaluate them in that order.

3 Setting and Notation

In this section, we formally introduce the repeated decision-making protocol for an agent whose goal is to maximize the total return on investment in a sequence of decision tasks.

The only two choices that an agent makes in a decision task are when to stop gathering information on the current innovation and whether or not to accept the innovation based on this information. In other words, the behavior of the agent during each task is fully characterized by the choice of a pair $\pi = (\tau, \mathrm{accept})$ that we call a *(decision-making) policy* (for the interested reader, Section A of the Supplementary Material contains a short mathematical discussion on policies), where:

- $\tau(\boldsymbol{x})$, called *duration*, maps a sequence of observations $\boldsymbol{x} = (x_1, x_2, \ldots)$ to an integer d (the no. of observations after which the learner stops gathering info on the current innovation);
- $\mathrm{accept}(d, \boldsymbol{x})$, called *decision*, maps the firs d observations of a sequence $\boldsymbol{x} = (x_1, x_2, \ldots)$ to a boolean value in $\{0, 1\}$ (where 1 represents accepting the current innovation).

An instance of our repeated decision-making problem is therefore determined by a set of admissible policies $\Pi = \{\pi_k\}_{k\in\mathcal{K}} = \{(\tau_k, \mathrm{accept})\}_{k\in\mathcal{K}}$ (with $\mathcal{K}$ finite or countable) and a distribution μ on $[-1, 1]$, modelling the value of innovations.[1] Naturally, the former is known beforehand but the latter is unknown and should be learned.

For a fixed choice of Π and μ, the protocol is formally described below. In each decision task n:

1. the *value* μ_n of the current innovation is drawn i.i.d. according to μ;
2. $\boldsymbol{X}_n$ is a sequence of i.i.d. (given μ_n) *observations* with $X_{n,i} = \pm 1$ and $\mathbb{E}[X_{n,i} \mid \mu_n] = \mu_n$;
3. the agent picks $k_n \in \mathcal{K}$ or, equivalently, a policy $\pi_{k_n} = (\tau_{k_n}, \mathrm{accept}) \in \Pi$;
4. the agent draws the first $d_n = \tau_{k_n}(\boldsymbol{X}_n)$ *samples*[2] of the sequence of observations $\boldsymbol{X}_n$;
5. on the basis of these sequential observations, the agent makes the decision $\mathrm{accept}\big(d_n, \boldsymbol{X}_n\big)$.

Crucially, μ_n is *never* revealed to the learner. We say that the agent *runs a policy* $\pi_k = (\tau_k, \mathrm{accept})$ (on a value μ_n) when steps 4–5 occur (with $k_n \leftarrow k$). We also say that they accept (resp., rejects) μ_n if their decision at step 5 is equal to 1 (resp., 0). Moreover, we say that the *reward* obtained and the *cost* payed by running a policy $\pi_k = (\tau_k, \mathrm{accept})$ on a value μ_n are, respectively,

$$\mathrm{reward}(\pi_k, \mu_n) = \mu_n \, \mathrm{accept}\big(\tau_k(\boldsymbol{X}_n), \boldsymbol{X}_n\big) \in \{\mu_n, 0\} \qquad \mathrm{cost}(\pi_k, \mu_n) = \tau_k(\boldsymbol{X}_n) \in \mathbb{N} \quad (1)$$

The objective of the agent is to converge to the highest ROI of a policy in Π, i.e., to guarantee that

$$R_N = \sup_{k\in\mathcal{K}} \frac{\sum_{n=1}^N \mathbb{E}\big[\mathrm{reward}(\pi_k, \mu_n)\big]}{\sum_{m=1}^N \mathbb{E}\big[\mathrm{cost}(\pi_k, \mu_m)\big]} - \frac{\sum_{n=1}^N \mathbb{E}\big[\mathrm{reward}(\pi_{k_n}, \mu_n)\big]}{\sum_{m=1}^N \mathbb{E}\big[\mathrm{cost}(\pi_{k_m}, \mu_m)\big]} \to 0 \quad \text{as } N \to \infty \quad (2)$$

where the expectations are taken with respect to μ_n, $\boldsymbol{X}_n$, and (possibly) the random choices of k_n.

To further lighten notations, we denote the expected reward, cost, and ROI of a policy π by

$$\mathrm{reward}(\pi) = \mathbb{E}\big[\mathrm{reward}(\pi, \mu_n)\big], \; \mathrm{cost}(\pi) = \mathbb{E}\big[\mathrm{cost}(\pi, \mu_n)\big], \; \mathrm{ROI}(\pi) = \mathrm{reward}(\pi)/\mathrm{cost}(\pi) \quad (3)$$

respectively and we say that $\pi_{k^\star}$ is an *optimal policy* if $k^\star \in \mathrm{argmax}_{k\in\mathcal{K}} \mathrm{ROI}(\pi_k)$. Note that $\mathrm{reward}(\pi)$ and $\mathrm{cost}(\pi)$ do not depend on n because μ_n is drawn i.i.d. according to μ.

For each policy $(\tau, \mathrm{accept}) \in \Pi$ and all tasks n, we allow the agent to reject the value μ_n regardless of the outcome of the sampling. Formally, the agent can always run the policy $(\tau, 0)$, where the second component of the pair is the decision identically equal to zero (i.e., the rule "always reject").

We also allow the agent to draw arbitrarily many extra samples in addition to the number $\tau(\boldsymbol{X}_n)$ that they would otherwise draw when running a policy $(\tau, \mathrm{accept}) \in \Pi$ on a value μ_n, provided that these additional samples are not taken into account in the decision to either accept or reject μ_n. Formally, the agent can always draw $\tau(\boldsymbol{X}_n) + k$ many samples (for any $k \in \mathbb{N}$) before making the decision $\mathrm{accept}\big(\tau(\boldsymbol{X}_n), \boldsymbol{X}_n\big)$, where we stress that the first argument of the decision function accept is $\tau(\boldsymbol{X}_n)$ and not $\tau(\boldsymbol{X}_n) + k$. Oversampling this way worsens the objective and might seem utterly counterproductive, but it will be crucial for recovering unbiased estimates of μ_n.

[1] We assume that the values of the innovations and the learner's observations belong to $[-1, 1]$ and $\{-1, 1\}$ respectively. We do this merely for the sake of readability (to avoid carrying over awkward constants or distributions $\mathcal{D}_n$). With a standard argument, both $[-1, 1]$ and $\{-1, 1\}$ can be extended to arbitrary codomains straightforwardly under a mild assumption of subgaussianity.

[2] Given μ_n, the random variable d_n is a stopping time w.r.t. the natural filtration associated to $\boldsymbol{X}_n$.

4 Competing Against K policies (CAPE)

As we mentioned in the introduction, in practice the duration of a decision task is defined by a capped early-stopping rule —e.g., drawing samples until 0 falls outside of a confidence interval around the empirical average, or a maximum number of draws has been reached. More precisely, if N tasks have to be performed, one could consider the natural policy class $\{(\tau_k, \mathrm{accept})\}_{k\in\{1,\dots,K\}}$ given by

$$\tau_k(\boldsymbol{x}) = \min\big(k,\ \inf\{d \in \mathbb{N} : |\overline{x}_d| \geq \alpha_d\}\big) \qquad \text{and} \qquad \mathrm{accept}(d, \boldsymbol{x}) = \mathbb{I}\{\overline{x}_d \geq \alpha_d\} \tag{4}$$

where $\overline{x}_d = (1/d)\sum_{i=1}^{d} x_i$ is the average of the first d elements of the sequence $\boldsymbol{x} = (x_1, x_2, \dots)$ and $\alpha_d = c\sqrt{(1/d)\ln(KN/\delta)}$, for some $c > 0$ and $\delta \in (0,1)$. While in this example policies are based on an Hoeffding concentration rule, in principle the learner is free to follow any scheme. Thus, we now generalize this notion and present an algorithm with provable guarantees against these finite families of policies.

Finite sets of policies. In this section, we focus on finite sets of K policies $\Pi = \{\pi_k\}_{k\in\{1,\dots,K\}} = \{(\tau_k, \mathrm{accept})\}_{k\in\{1,\dots,K\}}$ where accept is an arbitrary decision and $\tau_1, \dots, \tau_K$ is any sequence of bounded durations (say, $\tau_k \leq k$ for all k).[3] For the sake of convenience, we assume the durations are sorted by index ($\tau_k \leq \tau_h$ if $k \leq h$), so that τ_1 is the shortest and τ_K is the longest.

Divided we fall. A common strategy in online learning problems with limited feedback is explore-then-commit (ETC). ETC consists of two phases. In the first phase (explore), each action is played for the same amount of rounds, collecting this way i.i.d. samples of all rewards. In the subsequent commit phase, the arm with the best empirical observations is played consistently. Being very easy to execute, this strategy is popular in practice, but unfortunately, it is theoretically suboptimal in some applications. A better approach is performing action elimination. In a typical implementation of this strategy, all actions in a set are played with a round-robin schedule, collecting i.i.d. samples of their rewards. At the end of each cycle, all actions that are deemed suboptimal are removed from the set, and a new cycle begins. Neither one of these strategies can be applied directly because running a policy in our setting does not return an unbiased estimate of its reward (for a quick proof of this simple result, see Section F in the Supplementary Material). However, it turns out that we can get an i.i.d. estimate of a policy π by playing a *different* policy π'. Namely, one that draws *more* samples than π. This is reminiscent of bandits with a weakly observable feedback graph, a related problem for which the time-averaged regret over T rounds vanishes at a $T^{-1/3}$ rate (Alon et al., 2015). Albeit none of these three techniques works on its own, suitably interweaving all of them does.

United we stand. With this in mind, we now present our simple and efficient algorithm (Algorithm 1, CAPE) whose ROI converges (with high probability) to the best one in a finite family of policies. We will later discuss how to extend the analysis even further, including countable families of policies. Our algorithm performs policy elimination (lines 1–5) for a certain number of tasks (line 1) or until a single policy is left (line 6). After that, it runs the best policy left in the set (line 7) for all remaining tasks. During each policy elimination step, the algorithm oversamples (line 2) by drawing twice as many samples as it would suffice to take its decision $\mathrm{accept}\big(\tau_{\max(C_n)}(\boldsymbol{X}_n), \boldsymbol{X}_n\big)$ (at line 3). These extra samples are used to compute rough estimates of rewards and costs of all potentially optimal policies and more specifically to build *unbiased* estimates of these rewards. The test at line 4 has the only purpose of ensuring that the denominators $\widehat{c}_n^-(k)$ at line 5 are bounded away from zero so that all quantities are well-defined.

As usual in online learning, the *gap* in performance between optimal and sub-optimal policies is a complexity parameter. We define it as $\Delta = \min_{k\neq k^\star}\big(\mathrm{ROI}(\pi_{k^\star}) - \mathrm{ROI}(\pi_k)\big)$, where we recall that $k^\star \in \operatorname{argmax}_k \mathrm{ROI}(\pi_k)$ is the index of an optimal policy. Conventionally, we set $1/0 = \infty$.

Theorem 1. *If Π is a finite set of K policies, then the ROI of Algorithm 1 run for N tasks with exploration cap $N_{\mathrm{ex}} = \lceil N^{2/3} \rceil$ and confidence parameter $\delta \in (0,1)$ converges to the optimal $\mathrm{ROI}(\pi_{k^\star})$, with probability at least $1-\delta$, at a rate*

$$R_N = \widetilde{\mathcal{O}}\left(\min\left(\frac{K^3}{\Delta^2 N}, \frac{K}{N^{1/3}}\right)\right)$$

as soon as $N \geq K^3$ (where the $\widetilde{\mathcal{O}}$ notation hides only logarithmic terms, including a $\log(1/\delta)$ term).

[3] We chose $\tau_k \leq k$ for the sake of concreteness. All our results can be straightforwardly extended to arbitrary $\tau_k \leq D_k$ by simply assuming without loss of generality that $k \mapsto D_k$ is monotone and replacing k with D_k.

Algorithm 1: Capped Policy Elimination (CAPE)

Input: finite policy set Π, number of tasks N, confidence parameter δ, exploration cap N_{ex}
Initialization: let $C_1 \leftarrow \{1, \dots, K\}$ be the set of indices of all currently optimal candidates

1 **for** *task* $n = 1, \dots, N_{\text{ex}}$ **do**
2 draw the first $2\max(C_n)$ samples $X_{n,1}, \dots, X_{n,2\max(C_n)}$ of $\boldsymbol{X}_n$
3 make the decision $\text{accept}\big(\tau_{\max(C_n)}(\boldsymbol{X}_n), \boldsymbol{X}_n\big)$
4 **if** $n \geq 2K^2 \ln(4KN_{\text{ex}}/\delta)$ **then** let $C_{n+1} \leftarrow C_n \setminus C'_n$, where

5
$$C'_n = \big\{k \in C_n \,:\, \big(\widehat{r}_n^+(k) \geq 0 \text{ and } \widehat{r}_n^+(k)/\widehat{c}_n^-(k) < \widehat{r}_n^-(j)/\widehat{c}_n^+(j), \text{ for some } j \in C_n\big) \text{ or } \big(\widehat{r}_n^+(k) < 0 \text{ and } \widehat{r}_n^+(k)/\widehat{c}_n^+(k) < \widehat{r}_n^-(j)/\widehat{c}_n^-(j), \text{ for some } j \in C_n\big)\big\}$$

$$\widehat{r}_n^{\pm}(k) = \frac{1}{n}\sum_{m=1}^{n}\sum_{i=1}^{\max(C_m)} \frac{X_{m,\max(C_m)+i}}{\max(C_m)}\,\text{accept}\big(\tau_k(\boldsymbol{X}_m), \boldsymbol{X}_m\big) \pm \sqrt{\frac{2}{n}\ln\frac{4KN_{\text{ex}}}{\delta}} \quad (5)$$

$$\widehat{c}_n^{\pm}(k) = \frac{1}{n}\sum_{m=1}^{n}\tau_k(\boldsymbol{X}_m) \pm (k-1)\sqrt{\frac{1}{2n}\ln\frac{4KN_{\text{ex}}}{\delta}} \quad (6)$$

6 **if** $|C_{n+1}| = 1$ **then** let $\widehat{r}_{N_{\text{ex}}}^{\pm}(k) \leftarrow \widehat{r}_n^{\pm}(k)$, $\widehat{c}_{N_{\text{ex}}}^{\pm}(k) \leftarrow \widehat{c}_n^{\pm}(k)$, $C_{N_{\text{ex}}+1} \leftarrow C_{n+1}$, **break**
7 run policy $\pi_{k'}$ for all remaining tasks, where

$$k' \in \begin{cases} \operatorname*{argmax}_{k \in C_{N_{\text{ex}}+1}} \big(\widehat{r}_{N_{\text{ex}}}^+(k)/\widehat{c}_{N_{\text{ex}}}^-(k)\big) & \text{if } \widehat{r}_{N_{\text{ex}}}^+(k) \geq 0 \text{ for some } k \in C_{N_{\text{ex}}+1} \\ \operatorname*{argmax}_{k \in C_{N_{\text{ex}}+1}} \big(\widehat{r}_{N_{\text{ex}}}^+(k)/\widehat{c}_{N_{\text{ex}}}^+(k)\big) & \text{if } \widehat{r}_{N_{\text{ex}}}^+(k) < 0 \text{ for all } k \in C_{N_{\text{ex}}+1} \end{cases} \quad (7)$$

Proof sketch. This theorem relies on four technical lemmas (Lemmas 5-8) whose proofs are deferred to Section C of the Supplementary Material.

With a concentration argument (Lemma 5), we leverage the definitions of $\widehat{r}_n^{\pm}(k), \widehat{c}_n^{\pm}(k)$ and the i.i.d. assumptions on the samples $X_{n,i}$ to show that, with probability at least $1-\delta$, the event

$$\widehat{r}_n^-(k) \leq \text{reward}(\pi_k) \leq \widehat{r}_n^+(k) \qquad \text{and} \qquad \widehat{c}_n^-(k) \leq \text{cost}(\pi_k) \leq \widehat{c}_n^+(k) \quad (8)$$

occurs simultaneously for all $n \leq N_{\text{ex}}$ and all $k \leq \max(C_n)$. For the rewards, the key is oversampling, because $\text{accept}\big(\tau_k(\boldsymbol{X}_m), \boldsymbol{X}_m\big)$ in eq. (5) depends only on the first $k \leq \max(C_m)$ samples of $\boldsymbol{X}_m$ and is therefore independent of $X_{m,\max(C_m)+i}$ for all i. Assume now that (8) holds.

If $\Delta > 0$ (i.e., if there is a unique optimal policy), we then obtain (Lemma 6) that suboptimal policies are eliminated after at most N'_{ex} tasks, where $N'_{\text{ex}} \leq 288\,K^2 \ln(4KN_{\text{ex}}/\delta)/\Delta^2 + 1$. To prove it we show that a confidence interval for $\text{ROI}(\pi_k) = \text{reward}(\pi_k)/\text{cost}(\pi_k)$ is given by

$$\left[\frac{\widehat{r}_n^-(k)}{\widehat{c}_n^+(k)}\mathbb{I}\{\widehat{r}_n^+(k) \geq 0\} + \frac{\widehat{r}_n^-(k)}{\widehat{c}_n^-(k)}\mathbb{I}\{\widehat{r}_n^+(k) < 0\},\ \frac{\widehat{r}_n^+(k)}{\widehat{c}_n^-(k)}\mathbb{I}\{\widehat{r}_n^+(k) \geq 0\} + \frac{\widehat{r}_n^+(k)}{\widehat{c}_n^+(k)}\mathbb{I}\{\widehat{r}_n^+(k) < 0\}\right]$$

we upper bound its length, and we compute an N'_{ex} such that this upper bound is smaller than $\Delta/2$.

Afterwards, we analyze separately the case in which the test at line 6 is true for some task $N'_{\text{ex}} \leq N_{\text{ex}}$ and its complement (i.e., when the test is always false).

In the first case, by (8) there exists a unique optimal policy, i.e., we have that $\Delta > 0$. This is where the policy-elimination analysis comes into play. We can apply the bound above on N'_{ex}, obtaining a deterministic upper bound N''_{ex} on the number N'_{ex} of tasks needed to identify the optimal policy. Using this upper bound, writing the definition of R_N, and further upper bounding (Lemma 7) yields

$$R_N \leq \min\left(\frac{(2K+1)N_{\text{ex}}}{N},\ \frac{(2K+1)\big(288\,(K/\Delta)^2\ln(4KN_{\text{ex}}/\delta)+1\big)}{N}\right) \quad (9)$$

Finally, we consider the case in which the test at line 6 is false for all tasks $n \leq N_{\text{ex}}$, and line 7 is executed with $C_{N_{\text{ex}}+1}$ containing two or more policies. This is covered by a worst case explore-then-commit analysis. The key idea here is to use the definition of k' in Equation (7) to lower-bound $\text{reward}(\pi_{k'})$ in terms of $\text{reward}(\pi_{k^\star})/\text{cost}(\pi_{k^\star})$. This, together with some additional technical estimations (Lemma 8) leads to the result. □

5 Competing Against Infinitely Many Policies (ESC-CAPE)

Theorem 1 provides theoretical guarantees on the convergence rate R_N of CAPE to the best ROI of a finite set of policies. Unfortunately, the bound becomes vacuous when the cardinality K of the policy set is large compared to the number of tasks N. It is therefore natural to investigate whether the problem becomes impossible in this scenario.

Infinite sets of policies. With this goal in mind, we now focus on policy sets $\Pi = \{\pi_k\}_{k\in\mathcal{K}} = \big\{(\tau_k, \text{accept})\big\}_{k\in\mathcal{K}}$ as in the previous section, with $\mathcal{K} = \mathbb{N}$ rather than $\mathcal{K} = \{1, \ldots, K\}$.

We will show how such a countable set of policies can be reduced to a finite one containing all optimal policies with high probability (Algorithm 2, ESC). After this is done, we can run CAPE on the smaller policy set, obtaining theoretical guarantees for the resulting algorithm.

Estimating rewards and costs. Similarly to eqs. (5) and (6), we first introduce estimators for our target quantities. If at least $2k$ samples are drawn during each of n_2 consecutive tasks $n_1 + 1, \ldots, n_1 + n_2$, we can define, for all $\varepsilon > 0$, the following lower confidence bound on $\text{reward}(\pi_k)$:

$$\widehat{r}_k^-(n_1, n_2, \varepsilon) = \frac{1}{n_2} \sum_{n=n_1+1}^{n_1+n_2} \sum_{i=1}^{k} \frac{X_{n,k+i}}{k} \,\text{accept}\big(\tau_k(\boldsymbol{X}_n), \boldsymbol{X}_n\big) - 2\varepsilon \tag{10}$$

If at least $\tau_k(\boldsymbol{X}_n)$ samples are drawn during each of m_0 consecutive tasks $n_0 + 1, \ldots, n_0 + m_0$, we can define the following empirical average of $\text{cost}(\pi_k)$:

$$\overline{c}_k(n_0, m_0) = \big(\tau_k(\boldsymbol{X}_{n_0+1}) + \ldots + \tau_k(\boldsymbol{X}_{n_0+m_0})\big)/m_0 \tag{11}$$

A key observation. The key idea behind Algorithm 2 (ESC) is simple. Since all optimal policies $\pi_{k^\star}$ have to satisfy the relationships $\text{reward}(\pi_k)/\text{cost}(\pi_k) \leq \text{reward}(\pi_{k^\star})/\text{cost}(\pi_{k^\star}) \leq 1/\text{cost}(\pi_{k^\star})$, then, for all policies π_k with positive $\text{reward}(\pi_k)$, the cost of any optimal policy $\pi_{k^\star}$ must satisfy the relationship $\text{cost}(\pi_{k^\star}) \leq \text{cost}(\pi_k)/\text{reward}(\pi_k)$. In other words, *optimal policies cannot draw too many samples* and their cost can be controlled by estimating the reward and cost of *any* policy with positive reward.

We recall that running a policy $(\tau, 0)$ during a task n means drawing the first $\tau(\boldsymbol{X}_n)$ samples of $\boldsymbol{X}_n = (X_{n,1}, X_{n,2}, \ldots)$ and always rejecting μ_n, regardless of the observations.

Algorithm 2: Extension to Countable (ESC)

Input: countable policy set Π, number of tasks N, confidence parameter δ, accuracy levels $(\varepsilon_n)_n$
Initialization: for all j, let $m_j \leftarrow \big\lceil \ln\big(j(j+1)/\delta\big)/2\varepsilon_j^2 \big\rceil$ and $M_j = m_1 + \ldots + m_j$

1 **for** $j = 1, 2, \ldots$ **do**
2 run policy $\big(2 \cdot 2^j, 0\big)$ for m_j tasks and compute $\widehat{r}_{2^j}^- \leftarrow \widehat{r}_{2^j}^-(M_{j-1}, m_j, \varepsilon_j)$ as in (10)
3 **if** $\widehat{r}_{2^j}^- > 0$ **then** let $j_0 \leftarrow j$ and $k_0 \leftarrow 2^{j_0}$
4 **for** $l = j_0 + 1, j_0 + 2, \ldots$ **do**
5 run policy $\big(\tau_{2^l}, 0\big)$ for m_l tasks and compute $\overline{c}_{2^l} \leftarrow \overline{c}_{2^l}(M_{l-1}, m_l)$ as in (11)
6 **if** $\overline{c}_{2^l} > 2^l\,\varepsilon_l + k_0/\widehat{r}_{k_0}^-$ **then** let $j_1 \leftarrow l$ and **return** $K \leftarrow 2^{j_1}$

Thus, Algorithm 2 (ESC) first finds a policy π_{k_0} with $\text{reward}(\pi_{k_0}) > 0$ (lines 1–3), memorizing an upper estimate $k_0/\widehat{r}_{k_0}^-$ of the ratio $\text{cost}(\pi_{k_0})/\text{reward}(\pi_{k_0}) = 1/\text{ROI}(\pi_{k_0})$. By the argument above, this estimate upper bounds the expected number of samples $\text{cost}(\pi_{k^\star})$ drawn by *all* optimal policies $\pi_{k^\star}$. Then ESC simply proceeds to finding the smallest (up to a factor of 2) K such that $\text{cost}(\pi_K) \geq k_0/\widehat{r}_{k_0}^-$ (lines 4–6). Being $k_0/\widehat{r}_{k_0}^- \geq \text{cost}(\pi_{k_0})/\text{reward}(\pi_{k_0}) \geq \text{cost}(\pi_{k^\star})$ by construction, the index K determined this way upper bounds $k^\star$ for all optimal policies $\pi_{k^\star}$. (All the previous statements are intended to hold with high probability.) This is formalized in the following key lemma, whose full proof we defer to Section D of the Supplementary Material.

Lemma 2. *Let Π be a countable set of policies. If ESC is run with $\delta \in (0, 1)$, $\varepsilon_1, \varepsilon_2, \ldots \in (0, 1]$, and halts returning K, then $k^\star \leq K$ for all optimal policies $\pi_{k^\star}$ with probability at least $1 - \delta$.*

Before proceeding with the main result of this section, we need a final lemma upper bounding the expected cost of our ESC algorithm. This step is crucial to control the total ROI because in this

setting with arbitrarily long durations, picking the wrong policy even once is, in general, enough to drop the performance of an algorithm down to essentially zero, compromising the convergence to an optimal policy. This is another striking difference with other common online learning settings like stochastic bandits, where a single round has a negligible influence on the overall performance of an algorithm. To circumvent this issue, we designed ESC so that it tests shorter durations first, stopping as soon as the previous lemma applies, and a finite upper bound K on $k^\star$ is determined.

Lemma 3. *Let Π be a countable set of policies. If ESC is run with $\delta \in (0,1)$, $\varepsilon_1, \varepsilon_2, \ldots \in (0,1]$, and halts returning K, then the total number of samples it draws before stopping (i.e., its cost) is upper bounded by $\widetilde{\mathcal{O}}\big((K/\varepsilon^2)\log(1/\delta)\big)$, where $\varepsilon = \min\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_{\log_2 K}\}$.*

Proof. Note that, by definition, $\varepsilon = \min\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_{j_1}\} > 0$. Algorithm 2 (ESC) draw samples only when lines 2 or 5 are executed. Whenever line 2 is executed ($j = 1, \ldots, j_0$) the algorithm performs m_j tasks drawing $2 \cdot 2^j$ samples each time. Similarly, whenever line 5 is executed ($l = j_0 + 1, \ldots, j_1$) the algorithm draws at most 2^l samples during each of the m_l tasks. Therefore, recalling that $j_1 = \log_2 K$, the total number of samples drawn by ESC before stopping is at most

$$\sum_{j=1}^{j_0} 2 \cdot 2^j m_j + \sum_{l=j_0+1}^{j_1} 2^l m_l \le 2 \sum_{j=1}^{j_1} 2^j m_j \le 2 j_1 2^{j_1} \left\lceil \frac{1}{2\varepsilon^2} \ln \frac{j_1(j_1+1)}{\delta} \right\rceil \qquad \square$$

The ESC-CAPE algorithm. We can now join together our two algorithms obtaining a new one, that we call ESC-CAPE, which takes as input a countable policy set Π, the number of tasks N, a confidence parameter δ, some accuracy levels $\varepsilon_1, \varepsilon_2, \ldots$, and an exploration cap N_{ex}. The joint algorithm runs ESC first with parameters $\Pi, N, \delta, \varepsilon_1, \varepsilon_2, \ldots$. Then, if ESC halts returning K, it runs CAPE with parameters $\{(\tau_k, \text{accept})\}_{k \in \{1,\ldots,K\}}, N, \delta, N_{\text{ex}}$.

Analysis of ESC-CAPE. Since ESC rejects all values μ_n, the sum of the rewards accumulated during its run is zero. Thus, the only effect that ESC has on the convergence rate R_N of ESC-CAPE is an increment on the total cost in the denominator of its ROI. We control this cost by minimizing its upper bound in Lemma 3. This is not a simple matter of taking all ε_j's as large as possible. Indeed, if all the ε_j's are large, the **if** clause at line 3 might never be verified. In other words, the returned index K depends on ε and grows unbounded in general as ε approaches $1/2$. This follows directly from the definition of our lower estimate on the rewards (10). Thus, there is a trade-off between having a small K (which requires small ε_j's) and a small $1/\varepsilon^2$ to control the cost of ESC (for which we need large ε_j's). A direct computation shows that picking constant accuracy levels $\varepsilon_j = N^{-1/3}$ for all j achieves the best of both worlds and immediately gives our final result.

Theorem 4. *If Π is a countable set of policies, then the ROI of ESC-CAPE run for N tasks with confidence parameter $\delta \in (0,1)$, constant accuracy levels $\varepsilon_j = N^{-1/3}$, and exploration cap $N_{\text{ex}} = \lceil N^{2/3} \rceil$ converges to the optimal $\text{ROI}(\pi_{k^\star})$, with probability at least $1 - \delta$, at a rate*

$$R_N = \widetilde{\mathcal{O}}\left(\frac{1 + K\mathbb{I}\{\textit{ESC halts returning } K\}}{N^{1/3}} \right)$$

where the $\widetilde{\mathcal{O}}$ notation hides only logarithmic terms, including a $\log(1/\delta)$ term.

6 Conclusions and Open Problems

After formalizing the problem of ROI maximization in repeated decision making, we presented an algorithm (ESC-CAPE) that is competitive against infinitely large policy sets (Theorem 4). For this algorithm, we prove a convergence rate of order $1/N^{1/3}$ with high probability. To analyze it, we first proved a convergence result for its finite counterpart CAPE (Theorem 1), which is of independent interest. Notably, this finite analysis guarantees a significantly faster convergence of order $1/N$ on easier instances in which there is a positive gap in performance between the two best policies.

References

Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. 2012. Online prophet-inequality matching with applications to ad allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. Association for Computing Machinery, New York, NY, USA, 18–35.

Noga Alon, Nicolò Cesa-Bianchi, Ofer Dekel, and Tomer Koren. 2015. Online Learning with Feedback Graphs: Beyond Bandits. In *Proceedings of The 28th Conference on Learning Theory (Proceedings of Machine Learning Research, Vol. 40)*, Peter Grünwald, Elad Hazan, and Satyen Kale (Eds.). PMLR, Paris, France, 23–35.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47, 2-3 (2002), 235–256.

Eduardo M. Azevedo, Alex Deng, Jose Luis Montiel Olea, Justin Rao, and E. Glen Weyl. 2018. The A/B Testing Problem. In *Proceedings of the 2018 ACM Conference on Economics and Computation* (Ithaca, NY, USA) *(EC '18)*. Association for Computing Machinery, New York, NY, USA, 461–462. `https://doi.org/10.1145/3219166.3219204`

Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. 2018. Bandits with knapsacks. *Journal of the ACM (JACM)* 65, 3 (2018), 1–55.

Sébastien Bubeck and Nicolo Cesa-Bianchi. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning* 5, 1 (2012), 1–122.

Christopher F Chabris, Carrie L Morris, Dmitry Taubinsky, David Laibson, and Jonathon P Schuldt. 2009. The allocation of time in decision-making. *Journal of the European Economic Association* 7, 2-3 (2009), 628–637.

Shiyun Chen and Shiva Kasiviswanathan. 2020. Contextual Online False Discovery Rate Control. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 108)*, Silvia Chiappa and Roberto Calandra (Eds.). PMLR, Online, 952–961.

Jose Correa, Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. 2019. Recent developments in prophet inequalities. *ACM SIGecom Exchanges* 17, 1 (2019), 61–70.

Hossein Esfandiari, MohammadTaghi HajiAghayi, Brendan Lucier, and Michael Mitzenmacher. 2019. Online pandora's boxes and bandits. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (2019), 1885–1892.

P.W. Farris, N. Bendle, P.E. Pfeifer, and D. Reibstein. 2010. *Marketing Metrics: The Definitive Guide to Measuring Marketing Performance*. Pearson Education, New York.

Dale L. Flesher and Gary John Previts. 2013. Donaldson Brown (1885-1965): The power of an individual and his ideas over time. *The Accounting Historians Journal* 40, 1 (2013), 79–101.

Dean P Foster and Robert A Stine. 2008. α-investing: a procedure for sequential control of expected false discoveries. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70, 2 (2008), 429–444.

Christopher R Genovese, Kathryn Roeder, and Larry Wasserman. 2006. False discovery control with p-value weighting. *Biometrika* 93, 3 (2006), 509–524.

Thomas L Griffiths, Frederick Callaway, Michael B Chang, Erin Grant, Paul M Krueger, and Falk Lieder. 2019. Doing more with less: meta-reasoning and meta-learning in humans and machines. *Current Opinion in Behavioral Sciences* 29 (2019), 24–30.

Nicholas Hay, Stuart Russell, David Tolpin, and Solomon Eyal Shimony. 2012. Selecting Computations: Theory and Applications. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence* (Catalina Island, CA) *(UAI'12)*. AUAI Press, Arlington, Virginia, USA, 346–355.

Philipp Heesen and Arnold Janssen. 2016. Dynamic adaptive multiple tests with finite sample FDR control. *Journal of Statistical Planning and Inference* 168 (2016), 38–51.

Lalit Jain and Kevin Jamieson. 2018. Firing Bandits: Optimizing Crowdfunding. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholmsmässan, Stockholm Sweden, 2206–2214.

Adel Javanmard, Andrea Montanari, et al. 2018. Online rules for control of false discovery rate and false discovery exceedance. *The Annals of statistics* 46, 2 (2018), 526–554.

Ramesh Johari, Pete Koomen, Leonid Pekelis, and David Walsh. 2017. Peeking at A/B Tests: Why It Matters, and What to Do about It. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 1517–1525.

Robert Kleinberg, Bo Waggoner, and E. Glen Weyl. 2016. Descending Price Optimally Coordinates Search. In *Proceedings of the 2016 ACM Conference on Economics and Computation* (Maastricht, The Netherlands) *(EC '16)*. Association for Computing Machinery, New York, NY, USA, 23–24.

Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, and Nils Pohlmann. 2013. Online Controlled Experiments at Large Scale. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Chicago, Illinois, USA) *(KDD '13)*. Association for Computing Machinery, New York, NY, USA, 1168–1176.

Ang Li and Rina Foygel Barber. 2019. Multiple testing with the structure-adaptive Benjamini–Hochberg algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 81, 1 (2019), 45–74.

Brendan Lucier. 2017. An economic view of prophet inequalities. *ACM SIGecom Exchanges* 16, 1 (2017), 24–47.

Aaditya Ramdas, Fanny Yang, Martin J. Wainwright, and Michael I. Jordan. 2017. Online Control of the False Discovery Rate with Decaying Memory. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 5655–5664.

David S. Robertson and James M. S. Wason. 2018. Online control of the false discovery rate in biomedical research. arXiv:1809.07292 [stat.ME]

Dinah Rosenberg, Eilon Solan, and Nicolas Vieille. 2007. Social learning in one-arm bandit problems. *Econometrica* 75, 6 (2007), 1591–1611.

Sven Schmit, Virag Shah, and Ramesh Johari. 2019. Optimal Testing in the Experiment-rich Regime. In *Proceedings of Machine Learning Research (Proceedings of Machine Learning Research, Vol. 89)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.). PMLR, Naha, Okinawa, Japan, 626–633.

Aleksandrs Slivkins. 2019. Introduction to Multi-Armed Bandits. *Foundations and Trends® in Machine Learning* 12, 1-2 (2019), 1–286.

John Wilder Tukey. 1953. The Problem of Multiple Comparisons.

Abraham Wald. 1944. On cumulative sums of random variables. *The Annals of Mathematical Statistics* 15, 3 (1944), 283–296.

Martin L. Weitzman. 1979. Optimal Search for the Best Alternative. *Econometrica* 47, 3 (1979), 641–654.

Fanny Yang, Aaditya Ramdas, Kevin G Jamieson, and Martin J Wainwright. 2017. A framework for Multi-A(rmed)/B(andit) Testing with Online FDR Control. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, CA, USA, 5957–5966.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [N/A]
- Did you include the license to the code and datasets? [N/A]
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Introduction
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] We do not believe that a theoretical analysis could have any negative impacts on society
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Section 3 describes the setting, finite (resp., infinite) policy sets are introduced at the beginning of Section 4 (resp., Section 5), and everything else is included in the statements of lemmas and theorems
   (b) Did you include complete proofs of all theoretical results? [Yes] Some technical details are deferred to the Supplementary Material
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]