Additive Causal Bandits with Unknown Graph

Alan Malek [1] **Virginia Aglietti** [1] **Silvia Chiappa** [1]

Abstract

We explore algorithms to select actions in the causal bandit setting where the learner can choose to intervene on a set of random variables related by a causal graph, and the learner sequentially chooses interventions and observes a sample from the interventional distribution. The learner's goal is to quickly find the intervention, among all interventions on observable variables, that maximizes the expectation of an outcome variable. We depart from previous literature by assuming no knowledge of the causal graph except that latent confounders between the outcome and its ancestors are not present. We first show that the unknown graph problem can be exponentially hard in the parents of the outcome. To remedy this, we adopt an additional additive assumption on the outcome which allows us to solve the problem by casting it as an additive combinatorial linear bandit problem with full-bandit feedback. We propose a novel action-elimination algorithm for this setting, show how to apply this algorithm to the causal bandit problem, provide sample complexity bounds, and empirically validate our findings on a suite of randomly generated causal models, effectively showing that one does not need to explicitly learn the parents of the outcome to identify the best intervention.

1. Introduction

What setting of our factory production system should we choose to maximize efficiency? Which nutrients would induce maximal crop yield increase? What combination of drugs and dosages would optimize patients outcomes? All these questions ask which variables and values would optimize the *causal effect* on an outcome Y. In a system of variables $\boldsymbol{X}_{[K]} = \{X_1, \ldots, X_K\}$ and Y, these questions can be phrased as asking which set $\boldsymbol{X} \subseteq \boldsymbol{X}_{[K]}$ and values $\boldsymbol{x} \in \mathrm{supp}(\boldsymbol{X})$ would produce an optimal outcome under the *intervention* $\mathrm{do}(\boldsymbol{X} = \boldsymbol{x})$, whose effect is to alter the *observational distribution* $p(\boldsymbol{X}_{[K]}, Y)$ describing the existing relationships between $\boldsymbol{X}_{[K]} \cup Y$ by setting $\boldsymbol{X}$ to the fixed value $\boldsymbol{x}$. Indicating with $p(Y \,|\, \mathrm{do}(\boldsymbol{X} = \boldsymbol{x}))$ the distribution of Y under such an intervention, answering these questions is equivalent to solving the optimization problem $\max_{\boldsymbol{X} \subseteq \boldsymbol{X}_{[K]}, \boldsymbol{x} \in \mathrm{supp}(\boldsymbol{X}))} \mathbb{E}[Y \,|\, \mathrm{do}(\boldsymbol{X} = \boldsymbol{x})]$.

The causal bandit problem, proposed in Lattimore et al. (2016), is an extension of the multi-armed bandit problem to the setting where many variables can be intervened on and a causal graph $\mathcal{G}$ is used to describe the causal relationships among $\boldsymbol{X}_{[K]} \cup Y$. The learner solves this optimization problem by repeatedly choosing an intervention $(\boldsymbol{X}, \boldsymbol{x})$, also called an *action*, and observing a sample from $p(Y \,|\, \mathrm{do}(\boldsymbol{X} = \boldsymbol{x}))$. A naïve approach to the problem would be to treat each of the combinatorially many actions as independent and run a typical bandit algorithm. Instead, the causal graph enables us to reason about acting on sub-parts of the system and exploit the causal structure to reduce the action set $\mathcal{A}$. For example, consider a system of variables X_1, X_2, and Y with causal graph $X_1 \to X_2 \to Y$. Fact 1.1 below tells us that intervening on $\mathrm{pa}(Y)$, the *parents* of Y (or *direct causes*, i.e. the variables with an edge into Y), can always produce an expectation as high as the best intervention on any other set. As $\mathrm{pa}(Y) = \{X_2\}$, this means that the optimization problem can simplified to $\max_{\boldsymbol{x}_2 \in \mathrm{supp}(X_2)} \mathbb{E}[Y \,|\, \mathrm{do}(\boldsymbol{X}_2 = \boldsymbol{x}_2)]$.

Causal bandits have been explored under various assumptions on $\mathcal{G}$, $\mathcal{A}$, and the interventional distribution. Lattimore et al. (2016) assumed full knowledge of $\mathcal{G}$ and of the distribution of $\mathrm{pa}(Y)$ under any intervention in $\mathcal{A}$ and proposed an algorithm that selects all actions at the start to minimize a lower bound on sample complexity. Lu et al. (2020) proposed a UCB algorithm for the cumulative regret setting which exploits the observation that one does not need a confidence bound on individual actions but on how the actions affect $\mathrm{pa}(Y)$. Lu et al. (2021) was the first to considered an unspecified graph and instead considered the special case of a tree $\mathcal{G}$ with a singleton $\mathrm{pa}(Y)$, which allowed $\mathrm{pa}(Y)$ to be identified via binary search. De Kroon et al. (2022) did not place constraints on the structure of $\mathcal{G}$ and instead used causal discovery algorithms to learn a *separating set* (i.e. a

[1] DeepMind, London, UK. Correspondence to: Alan Malek <alanmalek@deepmind.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

set that d-separates variables we may intervene on from Y) which thereby allowed them to decompose the problem into learning the effect of the actions on the separating set and learning the distribution of Y conditioned on the separating set. This algorithm can be viewed as a generalization of previous causal bandit algorithms which effectively use $\mathrm{pa}(Y)$ as the separating set. Bilodeau et al. (2022) developed an algorithm that performs optimally when a given set is a separating set but fell back to a normal bandit algorithm otherwise. With the exception of Bilodeau et al. (2022), De Kroon et al. (2022), and Maiti et al. (2022), these works assumed no *latent confounders* (i.e. latent common causes), and many assumed that the learner could make arbitrary interventions on all variables excluding Y. Xiong & Chen (2023) were the first to consider the PAC (i.e. the fixed-confidence) setting and, similar to this work, provides instance-dependent PAC bounds. They study the known-graph case (potentially with unobserved variables), whereas we focus on the unknown-graph case with the additional assumptions that (i) there are no unobserved confounders between Y and its *ancestors*, and (ii) we may intervene on all variables in $\boldsymbol{X}_{[K]}$ (which is a common assumption in the causal bandit literature, see (Lee & Bareinboim, 2018; Lu et al., 2020)).

We refer to this problem as the *causal bandit with an unknown graph* (CBUG) problem. While the first assumption is self-explanatory and the third assumption common in the literature, our second assumption is a relaxation of the typical no-latent-confounders assumption and natural in several situations. For example, if one has a list that contains the parents of Y (e.g. built from domain experts or a noisy causal discovery algorithm), our assumption is still satisfied for arbitrary joint distributions on these variable (even with latent confounding between them and between other variables). Finally, this confounding assumption implies that the optimal intervention set is $\mathrm{pa}(Y)$, as demonstrated in Lee & Bareinboim (2018, Proposition 2).

Fact 1.1. *If there are no latent confounders between Y and any of its ancestors, then*

$$\max_{\boldsymbol{X}\subseteq\boldsymbol{X}_{[K]},\boldsymbol{x}\in\mathrm{supp}(\boldsymbol{X})} \mathbb{E}[Y \mid do(\boldsymbol{X}=\boldsymbol{x})] \leq \max_{\boldsymbol{x}'\in\mathrm{supp}(pa(Y))} \mathbb{E}[Y \mid do(pa(Y)=\boldsymbol{x}')].$$

This fact suggests that we can solve the CBUG problem by first learning $\mathrm{pa}(Y)$ then searching for the optimal values in $\mathrm{supp}(\mathrm{pa}(Y))$; we refer to this approach as *parents-first*. Since a *global intervention* $\mathrm{do}(\boldsymbol{X}_{[K]}=\boldsymbol{x})$ cuts all incoming edges into $\boldsymbol{X}_{[K]}$ in $\mathcal{G}$ such that only the edges from $\mathrm{pa}(Y)$ to Y remain, it suffices to use global interventions and avoid learning the parents.

Fact 1.2. *Under the conditions of Fact 1.1,*

$$\mathbb{E}[Y \mid do(\boldsymbol{X}_{[K]}=\boldsymbol{x})] = \mathbb{E}[Y \mid do(pa(Y)=\boldsymbol{x}')],$$

where $\boldsymbol{x}'=\boldsymbol{x}\cap\mathrm{supp}(pa(Y))$ are the values of $\boldsymbol{x}$ limited to $pa(Y)$.

Using the two above facts, the CBUG problem can be recast as a regression problem over $\mathrm{supp}(\boldsymbol{X}_{[K]})$. In other words, when there are no latent confounders between Y and any of its ancestors and when we may intervene on all variables, the optimal intervention may be found by a global intervention which circumvents the need to know the causal graph. To our surprise, this observation does not seem to have been used to design algorithms before.

Unfortunately, even with this simplification, the CBUG problem can be intractable. In Section 2, we show that, for any algorithm, there exists a problem instance with sample complexity exponential in the size of $\mathrm{pa}(Y)$; therefore, we need additional structural assumptions to make the problem tractable. Turning towards the causal inference literature, we see that the most common structural assumption is that the outcome is a noisy additive function of its parents, the discrete analog of the linear assumption. It has been developed into its own theory (Hastie, 2017) and is used throughout social and biomedical sciences: see e.g. Imbens & Rubin (2015, Chapter 13) and Bühlmann et al. (2014)), and more recently Maeda & Shimizu (2021) for examples. This assumption leads us to define the *additive* CBUG (aCBUG) problem, where we additionally assume that Y is an additive function of $\mathrm{pa}(Y)$ plus a random term. We emphasize that additive outcome settings are of significant interest to the causal inference community.

As developed in Section 3, the key implication of the additive assumption is that the aCBUG problem can be recast as a linear bandit problem (Lattimore & Szepesvári (2020, Chapter 19) provides a good introduction) where the action set is combinatorial and we only have full-bandit feedback, meaning that we never observe the effect of individual variables on the outcome and only observe a single sample of the outcome from $p(Y \mid \mathrm{do}(\boldsymbol{X}=\boldsymbol{x}))$.

By considering the specific problem of aCBUG, we have naturally arrived at the *additive combinatorial linear bandit problem with full-bandit feedback* problem. To the best of our knowledge, we are the first to consider this problem, which extends previous causal bandit settings in two ways: (1) the action set is combinatorial, whereas most prior pure-exploration linear bandits cannot scale to combinatorial actions, and (2) we only have full-bandit feedback, meaning that we can never observe the individual additive components of Y and must infer them from only their sum.

Existing pure-exploration linear bandit algorithms either have complexities that scale with the number of actions or cannot exploit the structure of the problem; hence, in Section 4, we propose a novel action-elimination algorithm that alternates between selecting actions that approximately solve an optimal design problem with decreasing tolerances and using the resulting observations to eliminate suboptimal actions. Noting that storing a combinatorial action set and solving optimal design problems are generally intractable, we solve both computational challenges by restricting $\mathcal{A}$ to *marginal* action sets that decomposes over variables, which also allows for an easy approximation of the optimal design problem. We name our algorithm *marginal optimal design linear bandit* (MODL). We analyze the algorithm in the PAC setting and provide one of the first instance-dependent analysis of the sample complexity (with Xiong & Chen (2023) being the only other, to be best of our knowledge). Finally, in Section 5, we show that MODL performs well for aCBUG problems and, in particular, significantly outperforms the parents-first approach while being only slightly behind an oracle version of the algorithm that knows $\text{pa}(Y)$.

2. Additive Causal Bandits with Unknown Graphs (aCBUG)

This section gives a formal definition of the CBUG problem, presents a lower bound showing that any algorithm for solving this problem must have exponential dependence on the number of parents, and introduces the additive outcome assumption.

Notation. We refer to sets of variables or values using bold face, e.g. $\boldsymbol{X}$ and $\boldsymbol{x}$, respectively. We use a subscript k to indicate variable number, superscripts i or j to indicate a discrete value, and superscripts n or t to indicate sample or round number. For example, $\boldsymbol{x}_k^t$ is the value of $\boldsymbol{X}_k$ for the tth sample, and θ_k^i is a parameter corresponding to X_k's ith value. Finally, $[n] := \{1, \ldots, n\}$ and $\boldsymbol{x}^{[n]}$ indicates a sequence of sets of values.

2.1. CBUG Problem Formulation

We assume a system formed by random variables $\boldsymbol{V} = \boldsymbol{X}_{[K]} \cup Y$, where $\boldsymbol{X}_{[K]} = \{X_1, \ldots, X_K\}$, $\text{supp}(X_k) = \{1, \ldots, M_k\}$ (i.e. each X_k has finite integer support), and Y is an outcome of interest that can be real-valued or discrete. The variables are causally related by an acyclic causal graph $\mathcal{G}$ with associated *observational distribution* $p(\boldsymbol{V})$. The learner acts by selecting a set $\boldsymbol{X} \subseteq \boldsymbol{X}_{[K]}$ and values $\boldsymbol{x} \in \text{supp}(\boldsymbol{X})$ and performing the *intervention* $\text{do}(\boldsymbol{X} = \boldsymbol{x})$, which corresponds to replacing $p(\boldsymbol{V})$ with the *interventional distribution* $p(\boldsymbol{V} \backslash \boldsymbol{X} \,|\, \text{do}(\boldsymbol{X} = \boldsymbol{x}))$ resulting from removing all incoming edges into $\boldsymbol{X}$ from $\mathcal{G}$ and fixing the value of $\boldsymbol{X}$ to $\boldsymbol{x}$. When there are no *unobserved confounders* (i.e. a latent common cause between variables in $\boldsymbol{V}$, usually represented by a bidirected edge in $\mathcal{G}$), $p(\boldsymbol{V}) = p(Y \,|\, \text{pa}(Y)) \prod_{k=1}^{K} p(X_k \,|\, \text{pa}(X_k))$, and the interventional distribution can be expressed as $p(\boldsymbol{V} \backslash \boldsymbol{X} \,|\, \text{do}(\boldsymbol{X} = \boldsymbol{x})) = p(Y \,|\, \text{pa}(Y)) \prod_{k=1 \text{s.t.} X_k \notin \boldsymbol{X}}^{K} p(X_k \,|\, \text{pa}(X_k)) \delta_{\boldsymbol{X}=\boldsymbol{x}}$ with $\delta_{\boldsymbol{X}=\boldsymbol{x}}$ a delta function centered at $\boldsymbol{x}$.

The learner's goal in causal bandits is to find the set $\boldsymbol{X}$ and values $\boldsymbol{x}$ which result in the greatest expectation of Y under the interventional distribution, denoted $\mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X} = \boldsymbol{x})]$. The learner accomplishes this task by interacting with the environment sequentially, choosing, at every round t, an intervention $(\boldsymbol{X}^t, \boldsymbol{x}^t)$ (also called an *action*) and obtaining a sample from $p(Y \,|\, \text{do}(\boldsymbol{X}^t = \boldsymbol{x}^t))$. Without loss of generality, we assume $\text{pa}(Y) = \{X_1, \ldots, X_{\mathcal{P}_Y}\}$ where $\mathcal{P}_Y$ is the number of parents of Y (of course the learner does not know this ordering).

We consider the causal bandit problem in the setting where $\mathcal{G}$ is unknown and the learner must find, for $\epsilon > 0$ and $\delta \in (0, 1)$, an (ϵ, δ)-PAC solution $(\hat{\boldsymbol{X}}, \hat{\boldsymbol{x}})$ satisfying $\mathbb{P}\Big(\max_{\boldsymbol{X}, \boldsymbol{x}} \mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X} = \boldsymbol{x})] - \mathbb{E}[Y \,|\, \text{do}(\hat{\boldsymbol{X}} = \hat{\boldsymbol{x}})] \leq \epsilon \Big) \geq 1 - \delta$. in as few rounds (or samples, we use the two interchangeably) as possible. We refer to this problem as the *causal bandit with an unknown graph* (CBUG) problem.

2.2. Global Intervention Approach

Recall that the CBUG problem assumes no latent confounders between Y and any of its *ancestors* (i.e. variables with a *directed (or causal) path* into Y) and that we can simultaneously intervene on all observable variables except for Y, i.e. we can make *global interventions*. A discussed previously, global interventions are a common model in the literature. They also model settings where statistical units are expensive relative to the cost of intervening on additional variables for a single unit, which implies that the sample complexity (the number of units used) is the key quantity to minimize, not the number of variables intervened on.

As stated in Fact 1.1, these two assumptions imply that $\text{pa}(Y)$ is the optimal intervention set. Thus, a natural solution to solving the CBUG problem would be to first find $\text{pa}(Y)$ and then search for the optimal value in $\text{supp}(\text{pa}(Y))$. Instead, we make the key observation that a *global intervention* $\text{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x})$ cuts all incoming edges into $\boldsymbol{X}_{[K]}$ in $\mathcal{G}$, leaving only the edges from $\text{pa}(Y)$ to Y, which implies that performing a global intervention is equivalent to intervening on $\text{pa}(Y)$; precisely, $\mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x})] = \mathbb{E}[Y \,|\, \text{do}(\text{pa}(Y) = \boldsymbol{x}')]$, where $\boldsymbol{x}'$ are the values of $\boldsymbol{x}$ for $\text{pa}(Y)$ (recall Fact 1.2 above). This claim can be proved by invoking rule 3 of do-calculus (Pearl, 2000) which, in this case, states that $\mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X}_{[K]})] = \mathbb{E}[Y \,|\, \text{do}(\text{pa}(Y))]$ since $Y \perp\!\!\!\perp_{\mathcal{G}_{\bar{\boldsymbol{X}}_{[K]}}} \boldsymbol{X}_{[K]} \backslash \text{pa}(Y) \,|\, \text{pa}(Y)$, where $\mathcal{G}_{\bar{\boldsymbol{X}}_{[K]}}$ is the graph $\mathcal{G}$ with all incoming edges into $\boldsymbol{X}_{[K]}$ removed. We can therefore restrict the problem to finding $\hat{x} \in \text{supp}(\boldsymbol{X}_{[K]})$ that satisfies $\mathbb{P}\Big(\max_{\boldsymbol{x} \in \text{supp}(\boldsymbol{X}_{[K]})} \mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x})] - \mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X}_{[K]} = \hat{\boldsymbol{x}})] \leq \epsilon \Big) \geq 1 - \delta$, meaning that learning the parents, in some sense, is optional.

2.3. The CBUG Problem is Exponentially Hard

While using global interventions saves us from having to consider all intervention sets in the powerset of $\boldsymbol{X}_{[K]}$, the CBUG problem can be exponentially hard in $\mathcal{P}_Y$. For a fixed δ and ϵ, let $\boldsymbol{x}^* \in \text{supp}(\text{pa}(Y))$ be fixed and unknown, and let $\mathbb{E}[Y \mid \text{do}(\text{pa}(Y) = \boldsymbol{x})] = 0 + \epsilon\mathbb{1}\left\{\boldsymbol{x} = \boldsymbol{x}^*\right\}$: the expectation is flat except at a single value $\boldsymbol{x}^*$ where it is equal to ϵ. We can strengthen the example by choosing p such that, for any intervention $\text{do}(\boldsymbol{X}' = \boldsymbol{x}')$ with $\boldsymbol{x}' \supseteq \boldsymbol{x}^*$ (indicating that $\boldsymbol{x}'$ agrees with $\boldsymbol{x}^*$ in $\text{supp}(\text{pa}(Y))$) we have $p(\boldsymbol{X}_k = \boldsymbol{x}^*_k \mid \text{do}(\boldsymbol{X}' = \boldsymbol{x}')) = 0$ for all $\boldsymbol{X}_k \notin \boldsymbol{X}'$. Only interventions with $\boldsymbol{x}' \supseteq \boldsymbol{x}^*$ can provide information about $\boldsymbol{x}^*$, so this problem is difficult as the learner has to try actions blindly until one containing $\boldsymbol{x}^*$ is found. We assume that $Y \mid \text{do}(\boldsymbol{X} = \boldsymbol{x})$ is 1-sub-Gaussian for any intervention.

Obtaining an upper bound on this problem is easy. Consider the algorithm that picks a ordering all values $\boldsymbol{x}^1, \boldsymbol{x}^2 \ldots$ in $\text{supp}(\boldsymbol{X}_{[K]})$ uniformly at random. Beginning at $t = 1$, it collects $O\left(\frac{\sum_k \log(M_k)}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ samples from $p(Y \mid \text{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x}^t)$ and tests $\mathbb{E}[Y \mid \text{do}(\boldsymbol{X} = \boldsymbol{x})] \geq \epsilon$ against the null hypothesis $\mathbb{E}[Y \mid \text{do}(\boldsymbol{X} = \boldsymbol{x})] = 0$. If the null hypothesis is rejected, then $\boldsymbol{x}^t$ is optimal and the algorithm terminates; otherwise, the algorithm moves on to $t + 1$. The sample complexity is $O\left(|\text{supp}(\text{pa}(Y))| \frac{\sum_k \log(M_k)}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$: since $\mathbb{P}(\boldsymbol{x}^* \in \boldsymbol{x}^t) = 1/|\,\text{supp}(\text{pa}(Y))|$, we have to test, on average, $O\left(|\,\text{supp}(\text{pa}(Y))|\right)$ actions before stumbling upon one containing $\boldsymbol{x}^*$. This naïve algorithm matches the following lower bound with proof in Appendix C.

Theorem 2.1. *There is an instance of the CBUG problem such that any (ϵ, δ)-PAC algorithm must take $\Omega\left(\frac{|\text{supp}(pa(Y))|}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ samples in expectation.*

2.4. Additive Outcome Assumption

As the example from the previous section illustrates, a problem where information about the optimal intervention is hyper-localized (i.e. where one only learns about the optimal intervention by trying it) is information-theoretically difficult. Therefore, we need some assumptions to make the problem tractable and, as discussed in the introduction, turn to the additive assumption from the causal inference literature (Bühlmann et al., 2014).

Assumption 2.2 (Additive Outcome)**.** There exist functions $f_1, \ldots, f_{\mathcal{P}_Y}$ and a σ^2-sub-Gaussian random variable η such that $Y = \sum_{k=1}^{\mathcal{P}_Y} f_k(X_k) + \eta$.

This assumption implies that the causal effect of $\text{pa}(Y)$ on Y decomposes into the sum of individual effects from each parent, which results in an *additive* CBUG (aCBUG) problem. We focus on the homoscedastic case where η is an i.i.d. σ^2-sub-Gaussian random variable, by far the most common assumption in the literature.

3. Pure-Exploration Linear Bandits

This section defines the additive combinatorial linear bandit with full-bandit feedback problem and shows how aCBUG is a special case. With the additive outcome assumption, the CBUG problem can be cast as a pure-exploration linear bandit problem with a combinatorial action set $\mathcal{A}$. The linear bandit problem is a sequential decision problem where, at round t, the learner chooses an action $\boldsymbol{x}^t \in \mathcal{A}$ and observes $y^t = \langle \boldsymbol{x}^t, \theta^* \rangle + \epsilon^t$, where ϵ^t is a zero-mean σ^2-sub-Gaussian random variable and $\theta^* \in \mathbb{R}^d$ is a fixed but unknown parameter. The goal of the learner is to find an (ϵ, δ)-PAC action in a few rounds/samples as possible.

We cast aCBUG as a linear bandit problem using one-hot-encoding. For $k \in [K]$, let $e_k(i)$ be the ith unit vector in $\mathbb{R}^{M_k}$, and for $\boldsymbol{x} \in \text{supp}(\boldsymbol{X}_{[K]})$, define $e(\boldsymbol{x}) = (e_1(\boldsymbol{x}_1), \ldots, e_K(\boldsymbol{x}_K))$ to be the concatenation of the one-hot vectors, which produces a mapping from $\text{supp}(\boldsymbol{X}_{[K]})$ to $\mathbb{R}^d$ with $d := \sum_k M_k$. Defining the vector $\theta = (f_1(1), f_1(2), \ldots, f_K(M_K))$, we obtain $\mathbb{E}[Y \mid \text{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x})] = \langle \theta^*, e(\boldsymbol{x}) \rangle$. Therefore, the goal is to find $\arg\max_{\boldsymbol{x} \in \text{supp}(\boldsymbol{X}_{[K]})} \langle \theta^*, e(\boldsymbol{x}) \rangle$. Note that the terms of θ^* corresponding to $X_k \notin \text{pa}(Y)$ are zero, so θ^* is sparse. It is important to note that we only have full-bandit feedback because we observe $\langle \theta^*, e(\boldsymbol{x}) \rangle$ and not the individual f_k components. Even though the action set $\mathcal{A}$ has a combinatorial structure with size $\prod_{k=1}^{K} M_k$, as it is the Cartesian product of choosing one value for each variable, the additive assumption allows us to consider the dimension-d linear problem instead.

Casting aCBUG as a linear bandit problem enables us to borrow from the extensive literature on pure-exploration linear bandits. One successful approach has been to treat the action selection as a optimal experimental design problem: that is, selecting actions to reveal as much information about a hidden parameter vector estimated through regression. While these optimal design problems tend to be intractable except for special cases, we show how to approximate our action set to avoid these difficulties.

This approach was pioneered by Soare et al. (2014), who had the insight a that the optimal design problem should optimize for learning the gaps between actions. Improvements in sample complexity were made by Xu et al. (2018) and Tao et al. (2018) by using a different estimator and a different design approximation strategy, respectively, while Fiez et al. (2019) considered the more general problem of transductive experimental design. A survey of optimal design in linear bandits can be found in Lattimore & Szepesvári (2020, Chapter 22). Unfortunately, all of these algorithms have complexity that is linear in the number of actions and are therefor not tractable for our combinatorial action space. Another line of work, (Chen et al., 2014; Gabillon et al., 2011), had the same additive action structure as us but assumed semi-bandit feedback, i.e. where noisy observations of individual $f_k(x_i)$ are possible. Because we only observe $y^t = \sum_{k=1}^{\mathcal{P}_Y} f_k(\boldsymbol{x}_k^t) + \eta$, we are in the full-bandit setting and cannot use these algorithms either.

The only work we are aware of in the pure-exploration combinatorial linear bandit with full-bandit feedback setting is by Du et al. (2021), who claimed the first efficient algorithm for this setting. Their approach uses a pre-sampling step to select a subset of the actions of size $O(poly(d))$ and then runs the algorithm of (Constantinou & Dawid, 2017). The resulting algorithm is fairly complex (requiring multiple sub-procedures including an entropy mirror-descent stage) and requires finding a size-d subset of actions with rank d. In our case, the rank of any subset of actions is at most $d-1$ so we cannot use this algorithm. Further, their algorithm is general purpose and cannot fully exploit the structure of our action space. Hence, we created a new algorithm, introduced in the following section.

4. Marginal Optimal Design Linear Bandit

Given data $\{(\boldsymbol{x}^t, y^t)\}_{t=1}^n$, we need to learn about the unknown parameter vector θ^* in a way that lets us quantify the uncertainty. With $V_n = \sum_{t\le n} \boldsymbol{x}^t(\boldsymbol{x}^t)^\top$ denoting the data covariance and $V_n^\dagger$ its pseudoinverse, we use the ordinary least squares (OLS) estimator, $\hat{\theta} = V_n^\dagger \sum_{t\le n} \boldsymbol{x}^t y^t$, which has the following Azuma-style confidence interval for $\hat{\theta}$ (see, e.g. Soare et al. (2014)):

Lemma 4.1. *Let $\hat{\theta}$ be the OLS estimator calculated from data $\boldsymbol{x}^{[n]}$ with covariance matrix V_n. For any $z \in \mathbb{R}^d$, $\delta \in (0,1)$, and for σ^2-sub-Gaussian η,* $\mathbb{P}\left(\langle\hat{\theta} - \theta^*, z\rangle \ge \sqrt{2\sigma^2 \left\|z\right\|_{{V_n}^\dagger}^2 \log(1/\delta)}\right) \le \delta.$

This proof, as well as all other omitted proofs, are given in Appendix C. The lemma requires $\boldsymbol{x}^{[n]}$ to not be a function of the data. The main challenge in using this inequality is that we need to solve for a sequence of covariates to minimize the bound.

Following Lattimore & Szepesvári (2020, Chapter 22), we propose an action-elimination algorithm that proceeds in phases. Each phase begins with a set $\mathcal{S}$ of plausibly best actions and a desired tolerance γ. The algorithm chooses actions to optimize the upper bound in Lemma 4.1, then uses this guarantee to prune $\mathcal{S}$. The tolerance is then decreased before the next phase. Phases repeat until an optimal action is identified or all sub-ϵ actions have been removed. There are two main computational difficulties. First, the number of actions, $|\mathcal{A}| = \prod_k M_k$, is very large, which makes the pruning step potentially intractable. We need algorithms that scale with the exponentially smaller ambient dimension $d = \sum_k M_k$. Second, the action selection (known as an optimal design problem) is a combinatorial optimization problem and generally difficult. We solve both problems at once by limiting $\mathcal{S}$ to sets that decompose over variables, defined below.

Marginal Action Sets. We say that a set of actions $\mathcal{S} \subseteq \mathcal{A} = \operatorname{supp}(\boldsymbol{X}_{[K]})$ is *marginal* if there exist $\mathcal{S}_k \subseteq \operatorname{supp}(X_k)$, $k = 1, \ldots, K$ such that $\mathcal{S} = \{(j_1, j_2, \ldots, j_K) : j_1 \in \mathcal{S}_1, j_2 \in \mathcal{S}_2, \ldots, j_K \in \mathcal{S}_K\}$. In other words, S consists of the Cartesian product of $\mathcal{S}_1, \ldots, \mathcal{S}_K$. Marginal action sets are intuitive: if we have eliminated, say, $X_k = j$ as a good action, then we should never consider any action where $X_k = j$. Marginal action sets solve the combinatorial action set problem, since such sets can be represented by $\sum_k M_k$ binary values.

Optimal Design. In linear bandits, our goal is to choose actions $\boldsymbol{x}^t$ to reveal as much about the optimal action as possible. While the most obvious goal would be to choose actions to minimize a bound on $\langle\hat{\theta} - \theta^*, e(\boldsymbol{x})\rangle$ simultaneously for all actions $\boldsymbol{x} \in \mathcal{S}$, estimating the *gaps* between actions $\boldsymbol{x}$ and $\boldsymbol{x}'$, defined as $\Delta(\boldsymbol{x}, \boldsymbol{x}') := \langle\theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle$, is more efficient (Fiez et al., 2019). Defining $V_n := \sum_{t=1}^n e(\boldsymbol{x}^t) e(\boldsymbol{x}^t)^\top$, the optimal design problem is

$$\arg\min_{\boldsymbol{x}^{[n]}} \max_{\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}} \left\|e(\boldsymbol{x}) - e(\boldsymbol{x}')\right\|_{{V_n}^\dagger}. \tag{1}$$

Generally, the optimal design problem (1) is intractable (Xu et al., 2018), and the state-of-the-art algorithms are linear in $\mathcal{S}$ (Allen-Zhu et al., 2021). Fortunately, marginal action sets afford a computationally simple solution with an easy to calculate upper bound.

Lemma 4.2. *Assume that $\mathcal{S}$ is marginal, and let $\tilde{\boldsymbol{x}}^{[n]}$ be any sequence of actions that are uniform in every marginal, i.e. for every k, $\sum_{t=1}^n \mathbb{1}\{\boldsymbol{x}_k^t = i\} - \mathbb{1}\{\boldsymbol{x}_k^t = j\} \le 1$ for all $i, j \in \mathcal{S}_k$. With $\tilde{V}_n$ as the covariance matrix of $\tilde{\boldsymbol{x}}^{[n]}$, we have*

$$\max_{\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}} \left\|e(\boldsymbol{x}) - e(\boldsymbol{x}')\right\|_{\tilde{V}_n^\dagger} \le \sum_i \frac{2|\mathcal{S}_i|}{n - |\mathcal{S}_i|} \le \sum_i \frac{2|\mathcal{S}_i|}{n}.$$

Roughly, the proof proceeds by noting that $\tilde{V}_n$ can be written as a diagonal matrix of counts plus the cross terms, both of which are positive semi-definite. We can upper bound the total expression the norm defined only with the diagonal terms, which permits a particularly simple form of $\left\|e(\boldsymbol{x}) - e(\boldsymbol{x}')\right\|_{\tilde{V}_n^\dagger}$ that we can explicitly calculate for uniform sequences of marginals.

Remark 4.3. The embedding that we use for the linear bandits is not full rank. For example, for any vector $v \in \mathbb{R}^K$ with $\mathbf{1}^\top v = 0$, the null space of V_n includes

$(v_1\mathbf{1}(M_1), \ldots, v_K\mathbf{1}(M_K))$ (where $\mathbf{1}(n)$ is the ones vector of length n). However, what is important is that the projection of the nullspace onto the coordinates in $\mathcal{S}_k$ is always in the all-ones direction, which allows us to calculate unbiased estimates of the gaps, even though we may not be able to identify θ. This insight provides another reason why Eq. (1) is the correct optimization objective.

4.1. Deriving the Elimination Algorithm

At each phase of the algorithm, we have an action set $\mathcal{S}$ and an error tolerance γ, and we wish to find a set of actions R to remove from $\mathcal{S}$ that can be guaranteed to be suboptimal. The crux is that we can only calculate the empirical gaps $\hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}') \coloneqq \langle \hat{\theta}, e(\boldsymbol{x}) - e(\boldsymbol{x}') \rangle$, thus we need to bound the error $\hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}') - \Delta(\boldsymbol{x}, \boldsymbol{x}')$.

Suppose that we choose $\boldsymbol{x}^{[n]}$ according to Lemma 4.2; Lemma 4.1 then guarantees that $\langle \hat{\theta} - \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}') \rangle \leq \sqrt{4\sigma^2 \frac{\sum_k |\mathcal{S}_k|}{n} \log\left(\frac{1}{\delta}\right)}$ holds with high probability for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}$. This means that it suffices to take $n = \left\lceil \frac{4\sigma^2 |\sum_k \mathcal{S}_k|}{\gamma^2} \log\left(\frac{L}{\delta}\right) \right\rceil$ if we want to ensure that $\langle \hat{\theta} - \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}') \rangle \leq \gamma$.

The usual choice in the bandit literature is $R = \{\boldsymbol{x} \in \mathcal{S} : \exists \boldsymbol{x}' \in \mathcal{S} \text{ s.t. } \hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}') \geq \gamma\}$. Letting $\boldsymbol{x}^*$ be the optimal action, we see that

$$
\begin{aligned}
\langle \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}^*) \rangle &\leq \langle \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}') \rangle \\
&\leq \langle \hat{\theta}, e(\boldsymbol{x}) - e(\boldsymbol{x}') \rangle - \gamma \leq 0
\end{aligned}
$$

for all $\boldsymbol{x} \in R$. Using the inequality in the other direction, any $\boldsymbol{x} \in \mathcal{S} \setminus R$ must have $\langle \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}^*) \rangle \leq \langle \theta^* - \hat{\theta}, e(\boldsymbol{x}) - e(\boldsymbol{x}^*) \rangle + \langle \hat{\theta}, e(\boldsymbol{x}) - e(\boldsymbol{x}^*) \rangle \leq 2\gamma$.

This rejection procedure is guaranteed, with probability at least $1 - \delta$, to eliminate all 2γ-suboptimal actions and never eliminate the optimal action.

The reader may notice that $\mathcal{S} \setminus R$, is not marginal even with this choice of R, even if $\mathcal{S}$ is marginal. Instead, we want a R such that 1) $\mathcal{S} \setminus R$ is marginal, and 2) R is as large as possible. Such a marginal-preserving rejection rule is necessary for the tractability of Eq. (1).

Since we require $\mathcal{S}' \coloneqq \mathcal{S} \setminus R$ to be marginal, we can define $R_k \coloneqq \mathcal{S}_k \setminus \mathcal{S}'_k$ to be the values removed from X_k's marginal. How must we constrain R_k so that, for every $\boldsymbol{x} \in R$, there must be some $\boldsymbol{x}' \in \mathcal{S}$ with $\hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}') \geq \gamma$?

We use the fact that gaps decompose by variables: if we define $\hat{\Delta}_k^{i,j} \coloneqq \hat{\theta}_k^i - \hat{\theta}_k^j$ and $\hat{\Delta}_k^j \coloneqq \max_i \hat{\theta}_k^i - \hat{\theta}_k^j$, then the gap between $\boldsymbol{x}$ and $\boldsymbol{x}'$ decomposes as $\hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{k \in [K]} \hat{\Delta}_k^{\boldsymbol{x}_k, \boldsymbol{x}'_k}$. Taking $\boldsymbol{x}' = \boldsymbol{x}^*$, we have that $\hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}^*) = \sum_k \hat{\Delta}_k^{\boldsymbol{x}_k}$. Thus, we may only include $i \in R_k$ if, for all $\boldsymbol{x} \in \mathcal{S}$ with $\boldsymbol{x}_k = i$, we can guarantee that $\hat{\Delta}(\boldsymbol{x}, \boldsymbol{x}^*) = \sum_k \hat{\Delta}_k^{\boldsymbol{x}_k} \geq \gamma$. Using $\boldsymbol{x}^*(i)$ to denote $\boldsymbol{x}^*$ with the kth value set to i, it is easy to check that $\hat{\Delta}(\boldsymbol{x}^*(i), \boldsymbol{x}^*) = \hat{\Delta}_k^i$, which implies that we can only include i in R_k if $\hat{\Delta}_k^i \geq \gamma$.

Algorithm 1 MODL

```
Input: $\delta > 0$, $\epsilon > 0$, $\boldsymbol{X}_{[K]}$, $\sigma^2$, $B$
Optional: upper bound $\overline{\mathcal{P}}_Y$ on $\mathcal{P}_Y$
$\mathcal{S}_k(1) \leftarrow [M_k]$ for $k = 1, \ldots, K$
$L \leftarrow \left\lfloor \log\left(\frac{2BK}{\epsilon}\right) \right\rfloor$
for $\ell = 1, \ldots, L$ do
    $\gamma(\ell) \leftarrow \frac{\epsilon}{K} 2^{L-\ell+1}$
    $n \leftarrow \left\lceil \frac{4\sigma^2 |\sum_k \mathcal{S}_k(\ell)|}{\gamma(\ell)^2} \log\left(\frac{L}{\delta}\right) \right\rceil$
    Choose $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n \in \operatorname{supp}(\boldsymbol{X}_{[K]})$ using Lemma 4.2
    Collect $y^t \sim P(Y \mid \operatorname{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x}^t))\ \forall t \leq n$
    Update $V_n$, $\hat{\theta}(\ell) = V_n^\dagger \sum_{t \leq n} \boldsymbol{x}^t y^t$
    Calculate empirical gaps $\hat{\Delta}_k^j$
    for $k = 1, \ldots, K$ do
        $\mathcal{S}_k(\ell+1) \leftarrow \left\{ j \in \mathcal{S}_k(\ell) : \hat{\Delta}_k^j < \gamma(\ell) \right\}$
    end for
    $\hat{\mathcal{P}}_Y(\ell) \leftarrow \sum_k \mathbb{1}\{|\mathcal{S}_k(\ell)| = 1\}$
    if $\hat{\mathcal{P}}_Y(\ell+1) = K$ or $\hat{\mathcal{P}}_Y(\ell) \geq \overline{\mathcal{P}}_Y$ then
        Break
    end if
end for
Return $\arg\max_{\boldsymbol{x} \in \mathcal{S}} \langle \hat{\theta}(\ell), e(\boldsymbol{x}) \rangle$ and $\hat{\theta}(\ell)$
```

4.2. The MODL Algorithm

With the optimal design and rejection procedures derived, we can present the *marginal optimal design linear bandit* (MODL) algorithm and its sample complexity bound. MODL proceeds in phases $\ell = 1, \ldots, L$, and in each phase it solves an $\mathcal{XY}$-optimal design problem, using the results of Lemma 4.2 with error $\gamma = \epsilon 2^{L-\ell}$, ensuring that $\gamma = \frac{\epsilon}{2}$ by the time the algorithm terminates. The algorithm uses the rejection rule of Section 4.1 to maintain a marginal action set $\mathcal{S}$. We also consider the case when $\mathcal{P}_Y$ is provided, which allows termination once $\mathcal{P}_Y$ variables have their optimal value identified. The intuition is that $\theta_k^1, \ldots, \theta_k^{M_k}$ are approximately equal for all for $X_k \notin \operatorname{pa}(Y)$, so the algorithm is not able to limit $\mathcal{S}_k$ to a single value. Pseudocode is provided in Algorithm 1.

We remark that restricting to marginal action sets does not eliminate any action. Instead, this restriction potentially reduces the number of actions that can be eliminated by requiring that the set of remaining actions be expanded to the smallest marginal action set containing it. In essence, marginal action sets allow us to trade-off some statistical efficiency for computational tractability.

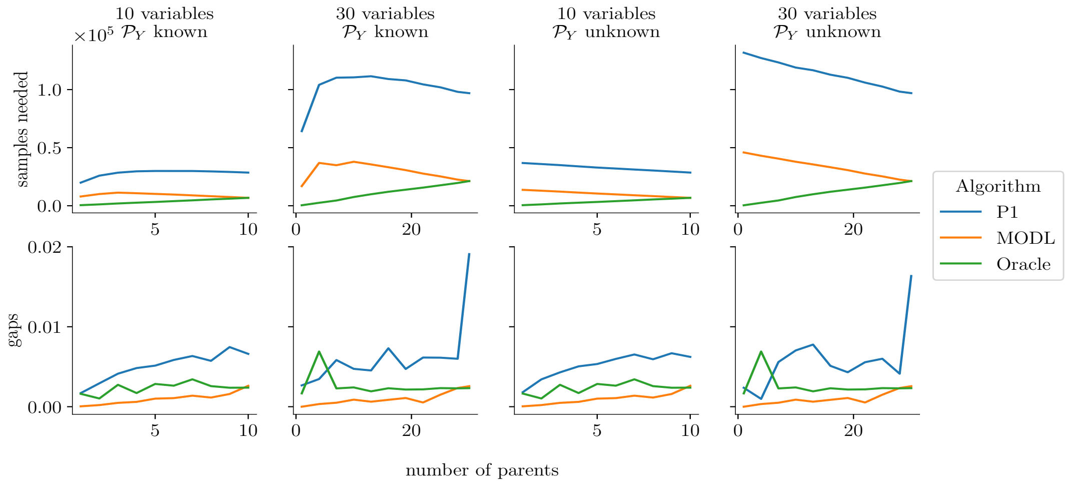

Figure 1. Sample complexity and average gaps versus number of parents of Y.

We analyzed the expected sample complexity of the algorithm and present an upper bound in Theorem 4.4. We find the typical sum-of-reciprocal-squared-gaps dependence common to best-arm-identification problems, $O\left(\sum_{i,k}(\Delta_k^i \vee \epsilon)^{-2}\right)$, however, instead of a sum over the combinatorial action set, the sum is over the all gaps for individual variables, which is the sample complexity one would expect if each variable could be played independently. In other words, despite only having full-bandit feedback, we obtain the sample complexity as if we had semi-bandit feedback. A substantial part of the complexity comes from the $\sum_{k \notin \text{pa}(Y)} M_k \epsilon^{-2}$ term, which arises because the non-parents are the most difficult: to differentiate between the cases when a variable is a non-parent or when there is a single value that is ϵ/K better than the rest, all values must be estimated within ϵ/K.

For the known-$\mathcal{P}_Y$ case, the $(\Delta_k^i \vee (\epsilon/K))^{-2}$ term is replaced by $(\Delta_k^i \vee \Delta_{\min} \vee (\epsilon/K))^{-2}$ which could be substantially smaller if $\Delta_{\min} \gg \epsilon/K$. We see this reduction because we no longer need to identify the non-parents, but rather can terminate once the minimum gap among the parents is found.

Theorem 4.4. *Algorithm 1 is (ϵ, δ)-PAC. The expected sample complexity has an upper bound of*

$$\frac{16\sigma^2}{3} \log\left(\frac{\log\left(\frac{BK}{\epsilon}\right)}{\delta}\right) \sum_{k=1}^{K} \sum_{i=1}^{M_k} \frac{1}{(\Delta_{\min} \vee \Delta_k^i \vee \frac{\epsilon}{K})^2},$$

where $\Delta_{\min} = \min_{k \le \mathcal{P}_Y} \min_{i \in [M_k]} \Delta_k^i$ is the minimum gap in the parents in the case when number of parents $\mathcal{P}_Y$ is provided, and 0 otherwise.

Due mostly to the additive assumption, the sample complexity contains $\sum_k M_k$ terms of order $O\left((\Delta \vee \epsilon)^{-2}\right)$, which is the same order as the sample complexity of running a separate bandit algorithm for each variable despite only observe the sum of rewards. In contrast, a naive approach which, ignoring the structure, simply uses a best-arm-identification algorithm over the combinatorial action set would have $\prod_k M_k$ terms in the complexity bound of order $\sum_{j_1=1}^{M_1} \cdots \sum_{j_K=1}^{M_K} \frac{1}{(\Delta_{\min} \vee (\sum_k \Delta_k^{j_k}) \vee \epsilon)^2}$.

Recovering the Parents of Y. With simple assumptions on f_k, we may recover a good estimate for $\text{pa}(Y)$ upon termination of the algorithm. Using the parameter estimates returned by the algorithm, we define $\widehat{\text{pa}}(Y)$ to be all the nodes X_k where $\forall \ell, |\hat{\theta}_k^i(\ell) - \hat{\theta}_k^j(\ell)| \le 2\gamma(\ell) \;\; \forall i, j \in \mathcal{S}_k(\ell)$. This formula follows the intuition that non-parents k have all θ_k^j identically equal and thus $\hat{\theta}_k^j(\ell)$ should be within the error tolerance $\gamma(\ell)$. We can show that this method works with high probability, provided an identifiability condition holds. Without any identifiability assumptions, no algorithm can be guaranteed to recover the parents.

Theorem 4.5. *Assume that there is some $\epsilon_{\min} > 0$ such that, for all $k \le \mathcal{P}_Y$, there exist $i, i' \in [M_k]$ with $|f_k(i) - f_k(i')| \ge \epsilon_{\min}$. Let $\hat{\boldsymbol{x}}$ and $\hat{\theta}$ be the output of Algorithm 1 run with $\epsilon \le \epsilon_{\min}$ and $\delta > 0$. Then $\widehat{pa}(Y)$ as defined above has $\mathbb{P}(\widehat{pa}(Y) = pa(Y)) \ge 1 - \delta$. Furthermore, the intervention $\hat{\boldsymbol{x}}_{\widehat{pa}(Y)} \coloneqq \{\boldsymbol{X}_i = \hat{\boldsymbol{x}}_i : \boldsymbol{X}_i \in \widehat{pa}(Y)\}$, which is $\hat{\boldsymbol{x}}$ limited to $pa(Y)$, is (ϵ, δ)-PAC.*

5. Experiments

This section presents an empirical evaluation of the MODL algorithm on a collection of randomly generated causal additive models[1]. Additional experiments studying the effect

[1] Code has been released at `https://github.com/deepmind/additive_cbug`.

of graph structure and the sensitivity to the additive outcome assumption's violation can be found in Appendix B.

Baselines. Since we are the first to consider the general setting of unknown $\mathcal{G}$ without assumptions on its structure, it is difficult to compare MODL to other algorithms in the causal bandit literature. The closest algorithms are those of De Kroon et al. (2022) and Bilodeau et al. (2022) with the separating set taken to be all intervenable random variables. In this settings, these algorithms reduces to a multi-armed bandit on the full, product action space. As the number of actions is exponential in the number of variables, we were only able to include this baseline for experiment with few variables. Since these algorithms were designed for the cumulative regret setting, we have implemented a version using Successive Elimination (SE) (Even-Dar et al., 2006).

We also compare MODL to (i) a *parents-first (P1) method* which first performs hypothesis testing to find an approximate parents set $\hat{\text{pa}}(Y)$ and then runs Algorithm 1 with $\boldsymbol{X}_{[K]} = \hat{\text{pa}}(Y)$ (i.e. considering $\hat{\text{pa}}(Y)$ as the intervention set), and (ii) an *oracle method* which runs Algorithm 1 with $\boldsymbol{X}_{[K]} = \text{pa}(Y)$. Fact 1.2 guarantees that intervening on $\text{pa}(Y)$ alone is sufficient to solve the problem; therefore, the difference in performance between MODL and the oracle method quantifies the value of knowing the parents. Comparing MODL with the P1 method answers whether spending samples to explicitly learn the parents is efficient.

For learning $\text{pa}(Y)$, we were not able to find any suitable algorithm in the literature that exploits the ability to intervene on all $\boldsymbol{X}_{[K]}$. Thus, we designed our own algorithm for finding an approximate parents set $\hat{\text{pa}}(Y)$ using global interventions. Let $\boldsymbol{x}_0 \in \text{supp}(\boldsymbol{X}_{[K]})$ be some fixed intervention; for each k in some random order, we enumerate $j \in [M_k]$ and test a null hypothesis of $\mathbb{E}[Y \,|\, \text{do}(\boldsymbol{X}_{[K]} = \boldsymbol{x}_0)] = \mathbb{E}[Y \,|\, \text{do}(X_k = j, \boldsymbol{X}_{[K]} \setminus \{X_k\} = \boldsymbol{x}_0)]$; X_k is added to $\hat{\text{pa}}(Y)$ only if we find a j where the null hypothesis is rejected. We terminate early if $\hat{\text{pa}}(Y)$ is large enough to meet a bound on $\mathcal{P}_Y$. Provided that, for all $X_k \notin \text{pa}(Y)$, there exists some j with $|f_k(j)| \geq \epsilon$, this algorithm is guaranteed to find $\text{pa}(Y)$ with high probability. Pseudocode and a complexity bound are provided in Appendix A.

Experimental Set-up. We performed the evaluation on randomly sampled structural causal models (SCMs) generated as followed. The causal graph, excluding Y, was a sampled directed acyclic graph from the Erdős-Rènyi model with the degree 3 and $K - 1$ variables. We randomly choose set of variables of size $\mathcal{P}_Y$ as the parents of Y, then each variable topologically greater than Y is independently set as a child to Y with probability .5.

To sample the conditional probability distributions, we chose M_k uniformly between specified upper and lower bounds and generated the conditional probability distribution for each X_k by sampling $p(X_k = j \,|\, \text{pa}(X_k) = \boldsymbol{x}) \propto \text{Beta}(2, 5)$ independently for all j and $\boldsymbol{x}$. Finally, we generated $f_k(j) = BW_k^j$, with $B = 5$ and W_k^j sampled i.i.d. from $\text{Beta}(2, 5)$ and set η to a standard normal variable. If X_j had Y as a parent, we used the same construction but with Y rounded to an integer.

Results. Using $\epsilon = 1/2$ and $\delta = .1$ for our (ϵ, δ)-PAC criterion, we considered a variety of settings of $\mathcal{P}_Y$, K, and upper and lower bounds for M_k. Each point all graphs corresponds to the average over 20 different SCMs sampled using the process described above and 50 independent runs of the methods on independently generated data.

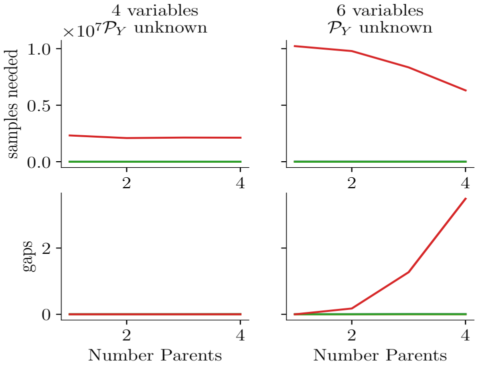

Figure 2. Sample complexities including SE.

In the figure above, the sample complexity and the average gaps are plotted for the Successive Elimination baseline (in red) as well as MODL, parents first, and the oracle methods. The SE baseline are almost too large to be comparable (roughly 200 times the other methods, all which appear comparatively as zero) and does not scale to more than a few variables. The sample complexity decreases with the number of parents as a greater portion of arms are able to be eliminated.

Figure 1 plots the same, without SE, for more interesting numbers of variables in four different combinations of $K = \{10, 30\}$ and known/unknown $\mathcal{P}_Y$ (the lower and upper bounds for M_k are 3 and 6). As predicted by Theorem 4.4, the sample complexity decreases with $\mathcal{P}_Y$: many samples are required to distinguish between non-parents and a potential parents with $\theta_k^i \approx \epsilon$, so the complexity increases with the number of non-parents. Overall, the performance of MODL is much closer to the performance of the oracle method. We see that the performance coincides when $\mathcal{P}_Y = K$ since MODL and the oracle method become the same algorithm. We also note that the P1 method does not benefit from $\mathcal{P}_Y = K$. The P1 method also has consistently higher gaps (since, on occasion, it fails to identify a parent which would cause a large error), but all gaps are well within the desired error tolerance of $\epsilon = 1/2$. See the appendix for more figures: e.g. Figure 3 plots the sample complexity for

$K = 30$ versus the support sizes M_k.

To summarize, we found that across all the settings that we investigated MODL was substantially better than the P1 method, and its performance (in terms of the gap of the final action and the sample complexity) was closer to the performance of the oracle method than to the performance of the P1 method. Hence, we conclude that the penalty of not knowing the parents is relatively small and much smaller than the cost of learning the parents first.

6. Discussion

In this paper, we have proposed an approach to solving the causal bandit problem under the general setting of an unknown causal graph (CBUG). Using the key insight that having no latent confounding between Y and any of its ancestors implies that a global intervention is equivalent to an intervention on the optimal set $\mathrm{pa}(Y)$, we showed that an additive outcome assumption allows us to solve the CBUG problem as a combinatorial linear bandit.

Limiting our algorithm to marginal action sets alleviated the computational burden by providing an easy approximation to the optimal design problem and a factorization of the action set. Two immediate direction for improving our algorithm and analysis is to consider the quality of approximation in the rejection procedure. A rejection procedure that outputs a marginal action set must reject fewer points than an unconstrained procedure. Are there principled ways of interpolating between marginal and full actions sets, perhaps using unions of marginal sets, which would let us trade-off computation and a larger number of rejected actions? The bound presented in this paper decomposes by variable, but a more nuanced rejection procedure and analysis should involve how the gaps between variables relate.

Other possible extensions include: (i) relaxing the additive outcome assumption, for example by adding "interaction terms," (ii) investigating what assumption allow for algorithms that adapt to sparsity, and (iii) considering the linear and continuous case. More generally, how can our method generalize to the case of latent confounders between Y and its ancestors?

Finally, we contrast our setting with the most common setting of causal inference where the goal is to learn the causal graph with as few experiments as possible. This setting is typical in the infinite data case, so the problem difficulty is measured in the number of experiments. As we are in the finite sample case, the number of statistical units (i.e. samples or rounds) is the most important quantity. With this distinction in mind, intervening on many variables is justified so long as we can reduce the sample complexity. However, budgeted versions of the causal bandit problem have been considered (Nair et al., 2021) and are another interesting direction.

Acknowledgements We wish to thank Eleni Sgouritsa for help in designing the code and technical support and Yasin Abbasi-Yadkori for technical feedback and many fruitful discussions.

References

Allen-Zhu, Z., Li, Y., Singh, A., and Wang, Y. Near-optimal discrete optimization for experimental design: A regret minimization approach. *Mathematical Programming*, 186:439–478, 2021.

Bilodeau, B., Wang, L., and Roy, D. M. Adaptively exploiting d-separators with causal bandits. In *Advances in Neural Information Processing Systems*, 2022.

Bühlmann, P., Peters, J., and Ernest, J. Cam: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6):2526–2556, 2014.

Chen, S., Lin, T., King, I., Lyu, M. R., and Chen, W. Combinatorial pure exploration of multi-armed bandits. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Constantinou, P. and Dawid, A. P. Extended conditional independence and applications in causal inference. *Annals of Statistics*, 45(6):2618–2653, 2017.

De Kroon, A., Mooij, J., and Belgrave, D. Causal bandits without prior knowledge using separating sets. In *Conference on Causal Learning and Reasoning*, pp. 407–427. PMLR, 2022.

Du, S. S., Kakade, S. M., Wang, R., and Yang, L. F. Is a good representation sufficient for sample efficient reinforcement learning? In *International Conference on Learning Representations*, 2020.

Du, Y., Kuroki, Y., and Chen, W. Combinatorial pure exploration with full-bandit or partial linear feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7262–7270, 2021.

Even-Dar, E., Mannor, S., Mansour, Y., and Mahadevan, S. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7(6):1079–1105, 2006.

Fiez, T., Jain, L., Jamieson, K. G., and Ratliff, L. Sequential experimental design for transductive linear bandits. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Gabillon, V., Ghavamzadeh, M., Lazaric, A., and Bubeck, S. Multi-bandit best arm identification. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

Hastie, T. J. Generalized additive models. In *Statistical models in S*, pp. 249–307. Routledge, 2017.

Imbens, G. W. and Rubin, D. B. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.

Lattimore, F., Lattimore, T., and Reid, M. D. Causal bandits: Learning good interventions via causal inference. In *Advances in Neural Information Processing Systems*, pp. 1181–1189, 2016.

Lattimore, T. and Szepesvári, C. *Bandit Algorithms*. Cambridge University Press, 2020.

Lee, S. and Bareinboim, E. Structural causal bandits: where to intervene? In *Advances in Neural Information Processing Systems*, pp. 2568–2578, 2018.

Lu, Y., Meisami, A., Tewari, A., and Yan, W. Regret analysis of bandit problems with causal background knowledge. In *Conference on Uncertainty in Artificial Intelligence*, pp. 141–150. PMLR, 2020.

Lu, Y., Meisami, A., and Tewari, A. Causal bandits with unknown graph structure. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24817–24828, 2021.

Maeda, T. N. and Shimizu, S. Causal additive models with unobserved variables. In *Uncertainty in Artificial Intelligence*, pp. 97–106, 2021.

Maiti, A., Nair, V., and Sinha, G. A causal bandit approach to learning good atomic interventions in presence of unobserved confounders. In *Uncertainty in Artificial Intelligence*, pp. 1328–1338. PMLR, 2022.

Nair, V., Patil, V., and Sinha, G. Budgeted and non-budgeted causal bandits. In *International Conference on Artificial Intelligence and Statistics*, pp. 2017–2025. PMLR, 2021.

Pearl, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

Soare, M., Lazaric, A., and Munos, R. Best-arm identification in linear bandits. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Tao, C., Blanco, S., and Zhou, Y. Best arm identification in linear bandits with linear dimension dependency. In *International Conference on Machine Learning*, pp. 4877–4886, 2018.

Xiong, N. and Chen, W. Combinatorial pure exploration of causal bandits. In *The Eleventh International Conference on Learning Representations*, 2023.

Xu, L., Honda, J., and Sugiyama, M. A fully adaptive algorithm for pure exploration in linear bandits. In *International Conference on Artificial Intelligence and Statistics*, pp. 843–851, 2018.

A. A simple test for the parents of Y

In the experiments, we compared MODL with an algorithm that first learns the parents of Y then ran a bandit algorithm on the remaining variables.

The algorithm is intuitively simple: we will construct confidence intervals of width $\epsilon/2$ for $\mathbb{E}[Y \mid \mathrm{do}(X_k = j, \boldsymbol{X}_{-k} = \boldsymbol{x}_{-k})]$ for all values of j, and if the intersection of the confidence intervals do not overlap, then one of the values of X_k is statistically significantly different from the others. Because we have intervened to fix all the other variables, this difference must be because X_k is a parent of Y. Formally, we have the following lemma.

Lemma A.1. *Let $\boldsymbol{x}_0 \in \mathrm{supp}(\boldsymbol{X})$ be fixed, and $\boldsymbol{x}_{-k}$ be $\boldsymbol{x}_0$ with the kth variable's value removed. For $\delta \in (0, 1)$ and $\epsilon > 0$, assume that $Y_k^1, \dots, Y_k^{n_k} \sim p(Y \mid do(X_k = j, \boldsymbol{X}_{-k} = \boldsymbol{x}_{-k})$, where*

$$n_k := \left\lceil \frac{8\sigma^2}{\epsilon^2} \log\left(\frac{2K \,\mathrm{supp}(X_k)}{\delta}\right) \right\rceil .$$

Then

$$\mathbb{P}\left(\forall 1 \le k \le K, j \in \mathrm{supp}(X_k), \left| \frac{\sum_{i=1}^{n_k} Y_k^i}{n_k} - \mathbb{E}[Y \mid do(X_k = j, \boldsymbol{X}_{-k} = \boldsymbol{x}_{-k})] \right| \le \frac{\epsilon}{2}\right) \ge 1 - \frac{1}{\delta}.$$

Hence, by the union bound, all the confidence intervals are simultaneously correct with probability at least $1 - \delta$.

Proof. With the n_k defined in the lemma, we can verify that

$$\sqrt{\frac{2\sigma^2}{n_k} \log\left(\frac{2K \,\mathrm{supp}(X_k)}{\delta}\right)} \le \frac{\epsilon}{2}.$$

Thus, Lemma 4.1 implies that

$$\mathbb{P}\left(\left| \frac{\sum_{i=1}^{n_k} Y_k^i}{n_k} - \mathbb{E}[Y \mid \mathrm{do}(X_k = j, \boldsymbol{X}_{-k} = \boldsymbol{x}_{-k})] \right| \le \frac{\epsilon}{2}\right) \ge 1 - \frac{2K \,\mathrm{supp}(X_k)}{\delta}.$$

Finally, by the union bound, all the confidence intervals are simultaneously correct with probability at least $1 - \delta$, as claimed. □

Under the event that the bounds are all correct, X_k is a parent of Y if there exist two intervals that do not overlap; in this case, the means of the two interventions must be different with high probability. See Algorithm 2 for pseudocode. In addition to having few false positives, we can argue that the algorithm has few false negatives, as presented in the following lemma.

Algorithm 2 Finding $\mathrm{pa}(Y)$

```
Given: $\epsilon > 0$, $\delta \in (0,1)$, $\sigma^2$, $\boldsymbol{x}_0 \in \mathrm{supp}(\boldsymbol{X}_{[K]})$ .
$\widehat{\mathrm{pa}}(Y) \leftarrow \emptyset$
for $k = 1, \dots, K$: do
    $C_k \leftarrow \mathbb{R}$
    $n \leftarrow \left\lceil \frac{8\sigma^2}{\epsilon^2} \log\left(\frac{2K\,\mathrm{supp}(X_k)}{\delta}\right)\right\rceil$ .
    for $j = 1, \dots, M_k$ do
        Collect $y^1, \dots, y^n \sim p(Y \mid \mathrm{do}(X_k = j, \boldsymbol{X}_{-i} = \boldsymbol{x}'))$
        $C_k \leftarrow C_k \cap \left(\frac{\sum_{i=1}^n y^i}{n} - \frac{\epsilon}{2}, \frac{\sum_{i=1}^n y^i}{n} + \frac{\epsilon}{2}\right)$
        if $C_k = \emptyset$ then
            $\widehat{\mathrm{pa}}(Y) \leftarrow \widehat{\mathrm{pa}}(Y) \cup \{X_k\}$
            Skip the rest of the tests for $X_k$.
        end if
    end for
end for
Return $\widehat{\mathrm{pa}}(Y)$
```

Lemma A.2. *Assume that for all $X_k \in pa(Y)$, there exists $i, j \in \mathrm{supp}(X_k)$ such that $|f_k(i) - f_k(j)| \ge \epsilon$. Then, with probability $1 - \delta$, Algorithm 2 correctly recovers the parents.*

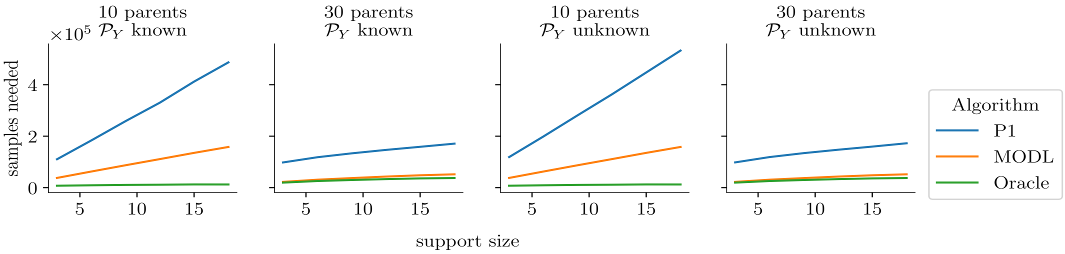

Figure 3. Sample complexity versus lower bound on M_k.

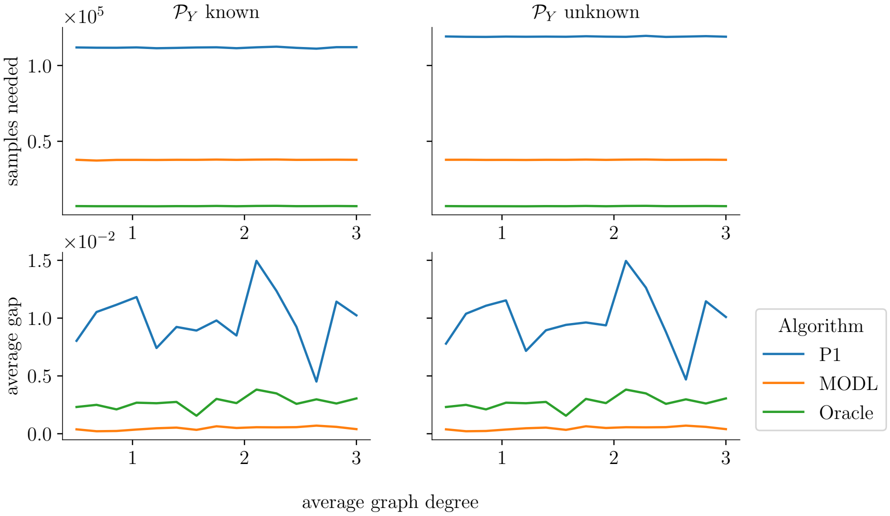

Figure 4. Sample complexity and average gap vs. number of variables with 30 variables and 10 parents.

B. Additional Experiments

This section holds additional experiments under the same setup in Section 5.

Figure 3 plots the sample complexity for $K = 30$ versus the support sizes M_k, where the x-axis specifies the lower bound on M_k (the upper bound is 3 larger). As above, the performance of MODL is much closer to the performance of the oracle method than to the performance of the P1 method, and almost identical for $\mathcal{P}_Y = K = 30$.

Figure 4 plots the sample complexity as the degree of the sampled graph changes, when $\mathcal{P}_Y = 10$ and $K = 30$. Unsurprisingly, the degree has very little effect on the sample complexity, confirming our intuition from causal inference that intervening on all parents renders the rest of the causal graph unimportant.

Figure 5 confirms our suspicion that the linear bandit is very sensitive to model mispecification. We generated non-linear data by using the outcome model

$$Y = \sum_{k=1}^{\mathcal{P}_Y} f_k(X_k) + \alpha B M_{\max}^{-4} \left(X_{i_1} X_{i_2} X_{i_3} X_{i_4} + X_{j_1} X_{j_2} X_{j_3} + X_{k_1} X_{k_2} \right)$$

for randomly chosen (without replacement) indices i_1, i_2, i_3, i_4, j_1, j_2, j_3, and k_1, k_2 from $[\mathcal{P}_Y]$ and $M_{\max}$ is the upper bound on the support size (6 in this case). This model was chosen to resemble the effect of adding “interaction terms” that are the product of several variables. The leading coefficient is chosen to keep the maximum interaction term roughly αB so choosing $\alpha \in [0, 1]$ keeps the scale of the interactions terms roughly equivalent to the additive terms. Despite this scaling, we still find that the performance is very sensitive to model mismatch, as illustrated in Figure 5. We also note that the parents-first approach is much more sensitive to model mispecification, at least in the average gap, because the mispecification increases the probability of the parents being misspecified, which in turn causes a large error. MODL and

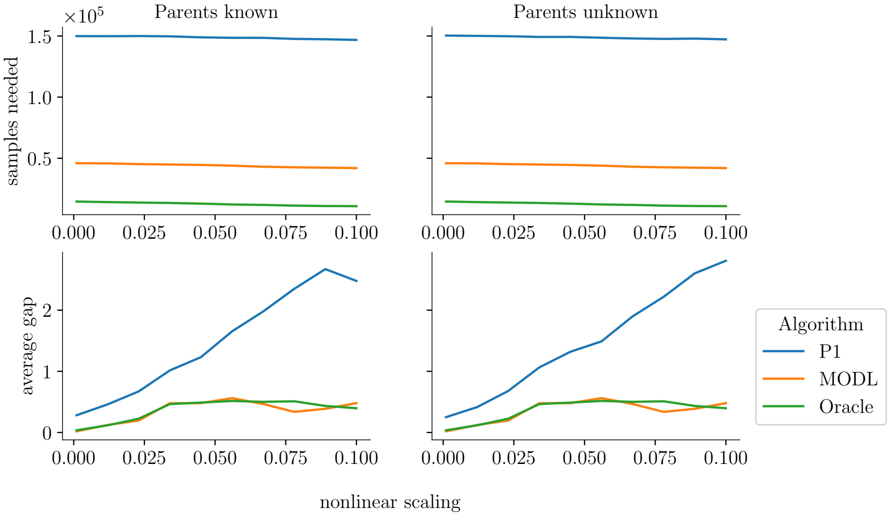

Figure 5. Sample complexity and average gap vs. model mispecification. The x-axis details the coefficient of a multiplicative nonlinear term in the expected response function.

the oracle algorithms are more immune to this effect.

C. Proofs

Proof of Theorem 2.1. This lower bound can by shown by invoking Theorem 33.5 from (Lattimore & Szepesvári, 2020).

For any $\boldsymbol{x} \in \operatorname{supp}(\boldsymbol{X}_{[K]})$, we define $q(\boldsymbol{x})$ to be the $\boldsymbol{x}$ values of $\operatorname{pa}(Y)$; in particular, the reward of any two actions $\boldsymbol{x}$ and $\boldsymbol{x}'$ are equal if $q(\boldsymbol{x}) = q(\boldsymbol{x}')$. We invoke the theorem for a bandit with $\operatorname{supp}(\boldsymbol{X}_{[K]})$ arms, one for each action, and with a set $\mathcal{E}$ of bandit environments indexed by $\boldsymbol{x}_{\text{pa}} \in \operatorname{supp}(\operatorname{pa}(Y))$; for a bandit $\nu(\boldsymbol{x}_p a) \in \mathcal{E}$ corresponding to $\boldsymbol{x}_{\text{pa}}$, we set $Y \,|\, \operatorname{do}(\boldsymbol{X} = \boldsymbol{x}) \sim \text{Bernoulli}(\frac{1}{2} + \epsilon \mathbb{1}\left\{q(\boldsymbol{x}) = \boldsymbol{x}_{\text{pa}}\right\})$. We can explicitly calculate $\mathcal{E}_{alt}(\nu(\boldsymbol{x}_{\text{pa}})) = \{\nu(\boldsymbol{x}'_{\text{pa}}) : \boldsymbol{x}'_{\text{pa}} \neq \boldsymbol{x}_{\text{pa}}\}$, which are the set of bandit environments with a different optimal arm than ν's. We also use ν_x to indicate the reward distribution of action $\boldsymbol{x}$ under bandit ν. Then, examining the terms in the theorem, we need to calculate

$$c(\nu) = \max_{\alpha \in \triangle_{\operatorname{supp}(\boldsymbol{X}_{[K]})}} \left(\min_{\nu' \in \mathcal{E}_{alt}(\nu)} \sum_{\boldsymbol{x} \in \operatorname{supp}(\boldsymbol{X}_{[K]})} \alpha_{\boldsymbol{x}} D(\nu_{\boldsymbol{x}}, \nu'_{\boldsymbol{x}}) \right),$$

where $D(\cdot,\cdot)$ is the relative entropy. If $\nu' \in \mathcal{E}_{alt}(\nu)$, this means that the $\boldsymbol{x}'_{\text{pa}}$ corresponding to ν' is different from the $\boldsymbol{x}_{\text{pa}}$ corresponding to ν. Fix one $\boldsymbol{x}'_{\text{pa}}$; we can see that $D(\nu_{\boldsymbol{x}}, \nu'_{\boldsymbol{x}})$ is zero for all $\boldsymbol{x}, \boldsymbol{x}'$ with $q(\boldsymbol{x}) = q(\boldsymbol{x}')$. We also use the fact that $D(\nu_{\boldsymbol{x}}, \nu'_{\boldsymbol{x}}) = O(\epsilon^2)$ when ϵ is small. Hence

$$\min_{\nu' \in \mathcal{E}_{alt}(\nu)} \sum_{\boldsymbol{x} \in \operatorname{supp}(\boldsymbol{X}_{[K]})} \alpha_{\boldsymbol{x}} D(\nu_{\boldsymbol{x}}, \nu'_{\boldsymbol{x}}) = \min_{\boldsymbol{x}'_{\text{pa}} \neq \boldsymbol{x}_{\text{pa}} \in \operatorname{supp}(\operatorname{pa}(Y))} O(\epsilon^2) \alpha_{\boldsymbol{x}'_{\text{pa}}},$$

where $\boldsymbol{x}_{\text{pa}}$ corresponds to ν. Taking the max over α, we see that any α must spread mass evenly across all $\boldsymbol{x}$ with $\boldsymbol{x}'_{\text{pa}} \neq \boldsymbol{x}_{\text{pa}} \in \operatorname{supp}(\operatorname{pa}(Y))$, which leads to $c(v) = O\left(\frac{\epsilon^2}{|\operatorname{supp}(\operatorname{pa}(Y))|}\right)$. Combining these calculations with the theorem, we find that

$$\mathbb{E}[\tau] \geq O\left(\frac{|\operatorname{supp}(\operatorname{pa}(Y))|}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right),$$

where $\mathbb{E}[\tau]$ is the expected stopping time of any sound (i.e. (ϵ,δ)-PAC) algorithm, as claimed.

We could also follow the techniques from the lower bound of (Du et al., 2020) and reduce the index-query problem to the CBUG problem. □

Proof of Lemma 4.2. Let $x^1, \dots, x^n$ be a sequence of actions. To calculate V_n, we write

$$V_n = \sum_{i=1}^{n} (e_1(\boldsymbol{x}_1^i), \dots, e_K(\boldsymbol{x}_K^i))(e_1(\boldsymbol{x}_1^i), \dots, e_K(\boldsymbol{x}_K^i))^\top = \sum_{i=1}^{n}\sum_{j=1}^{K}\sum_{j'=1}^{K} e_j(\boldsymbol{x}_j^i) e_{j'}(\boldsymbol{x}_{j'}^i) = D_n + C_n,$$

where D_n is the matrix of on-diagonal components and C_n contains all the off-diagonal terms; both are positive semi-definite. Using the fact that if A and B are PSD matrices, then $x^\top (A+B)^\dagger x \le x^\top A^\dagger x$, we can upper bound $\|e(\boldsymbol{x}) - e(\boldsymbol{x}')\|_{{V_n}^\dagger}$ by $\|e(\boldsymbol{x}) - e(\boldsymbol{x}')\|_{{D_n}^\dagger}$.

Letting $N_k^j = \sum_{t \le n} \mathbb{1}\left\{\boldsymbol{x}_k^t = j\right\}$ be the round $\boldsymbol{x}_k^t$ took on the jth value, we can show that

$$D_n = \sum_{t \le n} e(\boldsymbol{x}^t) e(\boldsymbol{x}^t)^\top = \sum_{t \le n} \operatorname{diag}((e_1(\boldsymbol{x}_1^t), \dots, e_K(\boldsymbol{x}_K^t)) = \operatorname{diag}\left(N_1^1, N_1^2, \dots, N_1^{M_1}, N_2^1, \dots, N_K^{M_K}\right);$$

D_n is the diagonal matrix of the counts of the values. We can upper bound the optimal design problem over $\boldsymbol{x}^{[n]}$ with another problem over counts of values that appears in $\mathcal{S}$ and add to n. Formally, this set is

$$\mathcal{N}(\mathcal{S}, n) := \left\{ (N_1^1, N_1^2, \dots, N_K^{M_K}) : \forall k, N_k^j = 0 \text{ if } j \notin \mathcal{S}_k \text{ and } \sum_j N_k^j = n \right\}.$$

We can easily check that, for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}$,

$$\begin{aligned} \|e(\boldsymbol{x}) - e(\boldsymbol{x}')\|_{{D_n}^\dagger} &= (e(\boldsymbol{x}) - e(\boldsymbol{x}'))^\top V_n^\dagger (e(\boldsymbol{x}) - e(\boldsymbol{x}')) \\ &= \sum_k (e_k(\boldsymbol{x}_k) - e_k(\boldsymbol{x}'_k))^\top \left(\frac{e_k(\boldsymbol{x}_k)}{N_k^{\boldsymbol{x}_k}} - \frac{e_k(\boldsymbol{x}'_k)}{N_k^{\boldsymbol{x}'_k}} \right) \\ &= \sum_k \left(\frac{1}{N_k^{\boldsymbol{x}_k}} + \frac{1}{N_k^{\boldsymbol{x}'_k}} \right) \mathbb{1}\left\{\boldsymbol{x}_k \ne \boldsymbol{x}'_k\right\}. \end{aligned}$$

Thus, the $\mathcal{XY}$-optimal has an upper bound

$$\begin{aligned} \boldsymbol{x}^{\mathcal{XY}}(\mathcal{S}, n) &= \arg \min_{\{N_i^j\} \in \mathcal{N}(\mathcal{S}, n)} \max_{\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}} \|e(\boldsymbol{x}) - e(\boldsymbol{x}')\|_{{V_n}^\dagger} \\ &= \arg \min_{\{N_k^j\} \in \mathcal{N}(\mathcal{S}, n)} \sum_k \max_{j, j' \in \mathcal{S}_k} \frac{1}{N_k^j} + \frac{1}{N_k^{j'}} \\ &= \arg \min_{\{N_k^j\} \in \mathcal{N}(\mathcal{S}, n)} \sum_k \frac{2}{\min_{j \in \mathcal{S}_k} N_k^j}. \end{aligned}$$

We can solve the problem in closed from: for all $i \le V$, allocate N_i^j evenly among all $j \in \mathcal{S}_i$. Because $|\mathcal{S}_i|$ may not divide n, we may have rounding errors and can only guarantee that $(N_i^j)^{-1} \in [\lfloor n/S_i \rfloor, \lceil n/S_i \rceil]$, which results in an objective value of $\sum_i \frac{2}{\lfloor n/S_i \rfloor} \le 2 \sum_i \frac{|S_i|}{n - |S_i|}$.

□

Theorem 4.4. *Algorithm 1 is (ϵ, δ)-PAC. The expected sample complexity is*

$$H^\epsilon := \frac{16}{3} \sigma^2 \log\left(\frac{\log(BK/\epsilon)}{\delta} \right) \left(\sum_{k \in pa(Y)} \sum_{i=1}^{M_k} \frac{1}{(\Delta_k^i \wedge (\epsilon/K))^2} + \sum_{k \notin pa(Y)} M_k \frac{1}{\epsilon^2} \right).$$

If the number of parents $\mathcal{P}_Y$ is provided, the complexity is instead

$$H^{\epsilon, \mathcal{P}_Y} = \frac{16}{3} \sigma^2 \log\left(\frac{\log(BK/\epsilon)}{\delta} \right) \sum_{k=1}^{K} \sum_{i=1}^{M_k} \frac{K^2}{(\Delta_{\min} \wedge \Delta_k^i \wedge (\epsilon/K))^2},$$

where $\Delta_{\min} = \min_{k \le \mathcal{P}_Y} \min_{i \in [M_k]} \Delta_k^i$ is the minimum gap in the parents.

Proof of Theorem 4.4. Recall that MODL alternates between two stages, data-collection and action elimination, and uses an exponentially decreasing error tolerance. Let $\mathcal{S}_k(\ell)$, $\gamma(\ell)$, and n_ell be the corresponding the values during phase ℓ.

It is easy to verify that $\gamma \approx B/2$ for $\ell = 1$ and $\gamma = \frac{\epsilon}{2K}$ for $\ell = L$. We will show that, with the chosen n_ℓ,

$$P\Big(\langle \hat{\theta} - \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle \leq \gamma(\ell) \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}(\ell), \ell \in [L]\Big) \geq 1 - \delta.$$

That is, with high probability, all our confidence intervals used by the algorithm are correct.

Let $\mathcal{S}_k(\ell)$ be the kth marginal of $\mathcal{S}$ during phase ℓ of the algorithm. By choosing

$$n_\ell = \left\lceil \frac{4\sigma^2 |\sum_k \mathcal{S}_k(\ell)|}{\gamma^2} \log\left(\frac{L}{\delta}\right) \right\rceil,$$

Lemma 4.2 guarantees that $\max_{z,z' \in \mathcal{S}} \|e(\boldsymbol{x}) - e(\boldsymbol{x}')\|_{V_n{}^\dagger} \leq \sum_k \frac{2|\mathcal{S}_k(\ell)|}{n}$. Lemma 4.1 provides

$$\langle \hat{\theta} - \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle \leq \sqrt{2\sigma^2 \|e(\boldsymbol{x}) - e(\boldsymbol{x}')\|_{V_n{}^\dagger} \log\left(\frac{L}{\delta}\right)} \leq \sqrt{\frac{4\sigma^2 \sum_k |\mathcal{S}_k(\ell)|}{n} \log\left(\frac{1}{\delta}\right)} \leq \gamma.$$

with probability at least $1 - \delta$ for all $\boldsymbol{x} \in \mathcal{S}(\ell)$ simultaneously.

At each stage ℓ, the elimination algorithm proceeds only eliminating $\boldsymbol{x}$ where there exists $\boldsymbol{x}'$ with $\langle \hat{\theta}, e(\boldsymbol{x}') - e(\boldsymbol{x})\rangle \geq \gamma(\ell)$. If such a $\boldsymbol{x}'$ exists, then we can conclude that

$$\langle \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}^*)\rangle \leq \langle \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle \leq \langle \hat{\theta}, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle - \gamma(\ell) \leq 0.$$

Hence, we have shown that, for all stages ℓ, the algorithm never eliminates any action that is $\gamma(\ell)$-suboptimal.

The last step to checking correctness is to show that an ϵ-suboptimal action is returned. In round $\ell = L$, Lemma 4.1, guarantees that $\langle \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle \leq \epsilon/2$ for all $\boldsymbol{x}, \boldsymbol{x}'$ in $\mathcal{S}(\ell)$. Applying this to the action $\hat{\boldsymbol{x}}$ returned by the algorithm and the fact that, under the event that the confidence intervals are correct, $\boldsymbol{x}^* \in \mathcal{S}(\ell)$, we have

$$\langle \theta^*, e(\hat{\boldsymbol{x}}) - e(\boldsymbol{x}^*)\rangle \leq \langle \theta^* - \hat{\theta}, e(\hat{\boldsymbol{x}}) - e(\boldsymbol{x}^*)\rangle + \langle \hat{\theta}, e(\hat{\boldsymbol{x}}) - e(\boldsymbol{x}^*)\rangle \leq \frac{K\epsilon}{2K} + \frac{\epsilon}{2},$$

where the last step used the fact that each vairable's error was controlled to $\epsilon/2k$.

The total sample complexity is

$$\sum_{\ell=1}^{L} \sum_{k=1}^{K} |\mathcal{S}_k(\ell)| \frac{4\sigma^2}{\gamma(\ell)^2} \log\left(\frac{L}{\delta}\right).$$

We now look to bound $\sum_{\ell=1}^{L} \sum_{k=1}^{K} |\mathcal{S}_k(\ell)|$ in terms of instance-dependent quantities. With no bound on $|\text{pa}(Y)|$, the complexity also decomposes. Let $G = \{\gamma_1 > \ldots > \gamma_L = \epsilon/2K\}$ be the set of γ used by the algorithm. Setting $c = 4\sigma^2 \log(L/\delta)$ and using the fact that $\hat{\Delta}_k^i$ are all unbiased, the expected sample complexity can be upper bounded by calculating

$$\begin{aligned}
\sum_{k=1}^{K} \sum_{\ell=1}^{L} \frac{c}{\gamma(\ell)^2} \left|\left\{\Delta_k^i \leq \gamma(\ell)\right\}\right| &= \sum_{k=1}^{K} \sum_{\gamma \in G} \sum_{i=1}^{M_k} \frac{c}{\gamma^2} \mathbb{1}\left\{\Delta_k^i \leq \gamma\right\} \\
&= \sum_{k=1}^{K} \sum_{i=1}^{M_k} \sum_{\{\gamma \in G : \gamma \geq \Delta_k^i \wedge \epsilon/K\}} \frac{c}{\gamma^2} \\
&\leq \sum_{k=1}^{K} \sum_{i=1}^{M_k} \sum_{j \geq 0} \frac{c}{2^{2j}\left(\Delta_k^i \wedge \frac{\epsilon}{K}\right)^2} = \sum_{k=1}^{K} \sum_{i=1}^{M_k} \frac{4c}{3\left(\Delta_k^i \wedge \frac{\epsilon}{K}\right)^2} \\
&= \sum_{k \in \text{pa}(Y)} \sum_{i=1}^{M_k} \frac{4c}{3(\Delta_k^i \wedge \frac{\epsilon}{K})^2} + \sum_{k \notin \text{pa}(Y)} M_k \frac{4cK^2}{3\epsilon^2}
\end{aligned}$$

To summarize, the sample complexity is the sum the sample complexity of the parents with a linear term for the non-parents. Intuitively, it is very difficult to differentiate between a parent with some $|f_k(\cdot)| \geq \epsilon$ and a non-parent, so we expect to see these terms in the sample complexity.

Recall that $\Delta_{\min} = \min_{k\in \mathrm{pa}(Y)} \min_{i\in[M_k]} \Delta_k^i$. When the number of parents is given, the algorithm will terminate as soon as $\gamma(\ell) \leq \Delta_{\min}$. Hence, the sample complexity can be decomposed into the samples needed to learn the parents, $\sum_{k\in\mathrm{pa}(Y)}\sum_{i=1}^{M_k}\frac{4c}{3(\Delta_k^i\wedge\epsilon)^2}$, and the extra samples from all the remaining variables

$$\begin{aligned}\sum_{\{\gamma\in G:\gamma\geq\Delta_{\min}\wedge\epsilon\}}\sum_{k\notin\mathrm{pa}(Y)} M_k\frac{c}{\gamma^2} &= \sum_{k\notin\mathrm{pa}(Y)} M_k c \sum_{\{\gamma\in G:\gamma\geq\Delta_{\min}\wedge\epsilon\}}\frac{1}{\gamma^2}\\ &\leq \sum_{k\notin\mathrm{pa}(Y)} M_k\frac{4c}{3(\Delta_{\min}\wedge\epsilon)^2}.\end{aligned}$$

Hence, the total sample complexity is

$$\frac{4c}{3}\sum_{k\in\mathrm{pa}(Y)}\sum_{i=1}^{M_k}\frac{1}{(\Delta_k^i\wedge\epsilon)^2} + \sum_{k\notin\mathrm{pa}(Y)}\frac{M_k}{\left(\Delta_{\min}\wedge\frac{\epsilon}{K}\right)^2} = \frac{4c}{3}\sum_{k=1}^{K}\sum_{i=1}^{M_k}\frac{1}{\left(\Delta_{\min}\wedge\Delta_k^i\wedge\frac{\epsilon}{K}\right)^2}.$$

□

Proof of Theorem 4.5. We need to prove two things: first, all parents are discovered, and second, no erroneous parents are included. Throughout, we condition of the event that the confidence intervals are all correct, which happens with probability at least $1-\delta$, and demonstrated in the proof of Theorem 4.4.

Recalling that $\hat{\theta}_k^i(\ell)$ and $\gamma(\ell)$ are the respective quantities at phase ℓ of the algorithm, we argue that non-parents are not added to $\hat{\mathrm{pa}}(Y)$. Consider some non-parent $k > \mathrm{pa}_Y$. The concentration inequality from Lemma 4.1, applied to $\boldsymbol{x}, \boldsymbol{x}'$ that only differ in that the first corresponds to $X_k = i$ and the second to $X_k = j$, yields

$$\langle \hat{\theta} - \theta^*, e(\boldsymbol{x}) - e(\boldsymbol{x}')\rangle \leq \gamma \Rightarrow \hat{\theta}_k^i - \theta_k^j \leq \gamma.$$

Taken with the fact that $\theta_k^i - \theta_k^j = 0$ for all $i, j \in M_k$, we have $\hat{\theta}_k^i(\ell) - \hat{\theta}_k^j(\ell) \leq 2\gamma(\ell)$, so k is not added to $\hat{\mathrm{pa}}(Y)$ on the event that the confidence intervals are collect.

To show that all parents are included, we consider two cases. First, assume that the algorithm does not terminate early so $\gamma_L = \epsilon/2$. For every $k \leq \mathcal{P}_Y$, the algorithm must have found the i, i' satisfying $|f_k(i) - f_k(i')| \geq \epsilon_{\min} \geq \epsilon/2$, so k in included in $\hat{\mathrm{pa}}_Y$.

The second case is when the algorithm terminates early. In this case, there must be $\overline{\mathcal{P}}_Y$ variables with only one action remaining. These variables must be parents, since, with high probability no non-parents are added.

Under the event that $\hat{\mathrm{pa}}(Y) = \mathrm{pa}(Y)$, Fact 1.2 implies that the expected response of actions $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{x}}_{\hat{\mathrm{pa}}(Y)}$ are equal, completing the proof. □