Algorithms for budget-constrained D-optimal design

Jiaqi Wang, Weijun Xie, Ilya O. Ryzhov

April 6, 2024

Abstract

D-optimal experimental design is a classical statistical problem in which one chooses a collection of data vectors, from some available large pool, in order to maximize a measure of predictive quality. In the classical formulation, the only constraint is on the cardinality of the collection, i.e., the number of vectors chosen. We study a more general budget-constrained variant in which vectors have heterogeneous costs, and develop four new algorithms (two deterministic, and two randomized) with approximation guarantees. Our methods handle heterogeneous costs using a novel exchange rule that interchanges packs of data vectors whose total costs are similar (up to some controlled amount of rounding error). The algorithms outperform the only existing method for this problem from both theoretical and empirical standpoints.

1 Introduction

Optimal experimental design is a class of statistical problems in which the data used to fit a regression model (such as ordinary least squares) are not exogenous, but can be chosen in advance. The goal is to obtain as much information as possible from a limited amount of data. Experimental design problems arise in the physical sciences (Sansone et al. 2019), engineering (Zhu et al. 2014), marketing research (Pokhilko et al. 2019), and other areas where data can be collected in a controlled setting; see Pukelsheim (2006) or Atkinson et al. (2007) for broad overviews of this field.

More specifically, let $\boldsymbol{v}_1, ..., \boldsymbol{v}_n$ denote n known vectors in $\mathbb{R}^d$ describing the settings of possible experiments; we use the notation $\{\boldsymbol{v}_i\}_{i\in[n]}$ for shorthand, denoting the index set by $[n] := \{1, 2, \ldots, n\}$. If the ith vector is selected, we will perform an experiment with settings $\boldsymbol{v}_i$ and collect a noisy observation of the quantity $\boldsymbol{v}_i^\top \boldsymbol{\beta}$, where $\boldsymbol{\beta}$ denotes regression coefficients. Thus, the vector $\boldsymbol{v}_i$ provides the covariates for a single data point, and the effects $\boldsymbol{\beta}$ can be estimated later from the entire set of selected experiments. The choice of experimental settings (i.e., the selection of $\boldsymbol{v}_i$) is made before any experimental outcomes are observed. Experiments may be conducted *without repetition* or *with repetition*, meaning that the same i may be chosen only once or multiple times.

In most practical contexts, experiments cost both time and money, and it is not possible to implement all n vectors. Instead, we are limited to some set (or multiset, if repetition is allowed)

$S \subseteq [n]$ of selected experiments. The literature (Madan et al. 2019, Singh & Xie 2020, Nikolov et al. 2022) generally assumes that experiments have the same cost, that is, the sole constraint is on the cardinality of S (sample size). In this paper, we study the more general *budget-constrained* variant of the problem, in which experiment i has a cost $c_i \geq 1$, and our selection decisions must satisfy the knapsack constraint $\sum_{i \in S} c_i \leq B$, where $B \in \mathbb{R}_+$ is a prespecified budget. While budget-constrained experimental design has received some attention (Horel et al. 2014, Ravi et al. 2016), existing work on algorithms with provable approximation guarantees is far more limited than for the cardinality-constrained case.

The experimental design literature has proposed many different objective functions for measuring the statistical quality of a dataset S. We focus on one of the most enduring and robust design objectives, namely the D-optimality criterion (de Aguiar et al. 1995). In the setting of ordinary least squares, D-optimal designs minimize the volume of the confidence ellipsoid for the regression parameters $\boldsymbol{\beta}$, which is equivalent to maximizing the deterministic objective $[\det(\sum_{i \in S} \boldsymbol{v}_i \boldsymbol{v}_i^\top)]^{\frac{1}{d}}$. Thus, we may formally state our problem as

$$\max_S \left\{ \left[\det \left(\sum_{i \in S} \boldsymbol{v}_i \boldsymbol{v}_i^\top \right) \right]^{\frac{1}{d}} : \sum_{i \in S} c_i \leq B, \mathrm{Supp}(S) \subseteq [n] \right\}, \tag{1}$$

where $\mathrm{Supp}(S)$ denotes the support of the (multi-)set S. We assume that there exists at least one set $\bar{S}$ with $\left|\bar{S}\right| = d$ for which $\sum_{i \in \bar{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top$ is nonsingular. Problem (1) depends only on the set S of selected vectors, not on the outcomes of any experiments; thus, we may use the D-optimality criterion to design the set ahead of time, before any experiments are conducted.

We develop four algorithms for problem (1). Two of them are deterministic, based on a notion of local search (LS), and denoted by LS-W and LS-WO for "with repetition" and "without repetition," respectively. The other two are based on random sampling (SA), and denoted analogously by SA-W and SA-WO. All four algorithms are polynomial-time. The first three (i.e., LS-W, LS-WO, and SA-W) are parameterized by two positive coprime integers $p > q$, as their analysis relies on the key observation that any cost c_i can be rounded up to the nearest integer power of the rational number $\frac{p}{q}$. The SA-WO algorithm relies on a weighted Chernoff bound and does not require any parameters. All four algorithms have provable multiplicative "α-approximation guarantees" of the

Table 1: In this table, $0 < \epsilon < 1$ is a small constant, $m := \lceil \log_{\frac{p}{q}}(c_{\max}) \rceil$, L is the maximum encoding length of $\boldsymbol{v}_i$ for all $i \in [n]$, δ is an improvement measure at each step of local search, and ϵ_0 is the precision of a root-finding step in RE.

	Approximation ratio	Threshold for B	Time	Full proof
LS-W	$(\frac{q}{p} - \epsilon)$	$q^m \frac{c_{\max} d}{\epsilon}$	$O(\delta^{-1}(Lnd^3(d^2 + m\log(p))B\log(B)))$	Theorem 2
LS-WO	$(\frac{q}{p} - \epsilon)$	$q^m \frac{c_{\max}(d+m+1)}{\epsilon}$	$O(\delta^{-1}(Lnd^3(d^2 + m\log(p))\log(B)))$	Theorem 4
SA-W	$\frac{1}{e}(\frac{q}{p} - \epsilon)$	$q^m \frac{c_{\max}}{\epsilon}$	$O(n^5 + n\lfloor \frac{B}{p^m} \rfloor)$	Theorem 5
SA-WO	$(1 - \epsilon)$	$\frac{4c_{\max} d}{\epsilon} + \frac{12c_{\max}}{\epsilon^2}\log(\frac{1}{\epsilon})$	$O(n^5)$	Theorem 6
RE (Lau & Zhou 2022)	$(1 - O(\epsilon))$	$\frac{2c_{\max} d}{\epsilon}(\epsilon < \frac{1}{200})$	$O(n^5 + \epsilon^{-1}d^2(d^3 + nd^2 + \log_2(\frac{\sqrt{d}-1}{\epsilon_0})))$	Theorem 1.4 in Lau & Zhou (2022)

form

$$\left[\det\left(\sum_{i\in S} \boldsymbol{v}_i \boldsymbol{v}_i^\top\right)\right]^{\frac{1}{d}} \geq \alpha \left[\det\left(\sum_{i\in S^*} \boldsymbol{v}_i \boldsymbol{v}_i^\top\right)\right]^{\frac{1}{d}},$$

where α is the *approximation ratio*, S is the dataset returned by the algorithm, and S^* is the optimal solution of (1). In all four cases, the approximation ratio is shown to hold when the budget B is above some explicit threshold. Table 1 summarizes our results for the four algorithms; for comparison, we also include the randomized exchange (RE) method of Lau & Zhou (2022), which to date is the only work with an approximation guarantee for budget-constrained D-optimal design. It is important to note that the RE method is not guaranteed to produce a feasible solution, and its guarantees hold only with some probability. RE also does not allow repetition.

In addition to the guarantees themselves, our analysis has several novel elements that could be of independent interest for the further study of nonlinear budget-constrained problems. In cardinality-constrained optimal design, local search methods (Li et al. 2024) work by interchanging, or swapping, one vector in S with one vector outside S to iteratively improve the objective. In budget-constrained optimal design, this approach does not work because the two vectors may have different costs. To handle this problem, we develop a novel potential function, using the superadditivity of the determinant, that allows us to interchange packs of vectors with "similar" costs that round to the same power of $\frac{p}{q}$. This technique drives the efficiency of LS-W and LS-WO. For the sampling algorithm SA-WO, we provide an efficient deterministic implementation that achieves the same approximation ratio using conditional expectations. Numerical experiments demonstrate that all four procedures are computationally efficient and can work even better in practice than what is guaranteed by theory.

This paper is organized as follows. Section 2 reviews the related literature. Sections 3-4 study

LS-W and LS-WO, whereas Sections 5-6 study SA-W and SA-WO. Each of the four algorithms requires some individual consideration and thus is placed in its own section. Section 7 presents numerical experiments, and Section 8 concludes.

2 Literature review

The literature on optimal experimental design is quite broad; we refer interested readers to Pukelsheim (2006) for a thorough discussion of the wide variety of optimality criteria that have been considered. As a subclass of nonlinear integer programming, optimal design problems can be approached using mathematical programming techniques. Ahipaşaoğlu (2021) proposed a general branch-and-bound algorithm that can exactly solve such problems while accommodating several of the most popular objectives. Sagnol & Harman (2015) focused more specifically on D-optimality and used mixed-integer second-order cone programming. Coey et al. (2023) characterized the log-determinant as a spectral function on Jordan algebras, allowing the use of conic optimization methods.

There is, by now, a fairly mature literature on approximation algorithms specifically for cardinality-constrained D-optimal design. A pipage rounding algorithm by Bouhtou et al. (2010) achieved a $\frac{k}{n}$-approximation, where k is the maximal number of experiments that can be selected. A local search algorithm by Madan et al. (2019), and a regret minimization approach by Allen-Zhu et al. (2021), both achieved a $(1-\epsilon)$-approximation. A sampling algorithm by Singh & Xie (2020) improved the result to a $1/e$-approximation in general and a $(1-\epsilon)$-approximation for large k. However, none of the analytical techniques in these papers can be directly extended to the budget-constrained problem, as the notion of interchange crucial to these local search methods becomes far less straightforward. To our knowledge, the only approximation guarantee for budget-constrained D-optimal design was obtained by Lau & Zhou (2022). The randomized exchange method presented in this work has a $(1-O(\epsilon))$-approximation; however, the method is not guaranteed to produce a feasible solution, and does not allow repetition.

A different stream of literature studies the more general problem class of budget-constrained maximization of submodular functions. For such problems, a modified greedy algorithm by Sviridenko (2004) achieved a $(1-e^{-1})$-approximation, while a "greedy+max" procedure by Yaroslavtsev et al. (2020) achieved a $(1/2-\epsilon)$-approximation. The work by Huang & Kakimura (2021) extended the problem to a streaming setting and derived a $(0.4-\epsilon)$-approximation algorithm. The

log-determinant is submodular, but there are some important nuances that limit the applicability of this work to our setting. First, the aforementioned papers all assume that the submodular function is nonnegative, monotone, and defined on all possible subsets. This is not the case in our setting, because the log-determinant can be negative and is not defined on singular matrices. Second, the α-approximations that we prove in this paper are for the function $\det^{1/d}(\cdot)$, which is consistent with existing literature on cardinality-constrained D-optimal design. Therefore, what we prove for the log-determinant is not a multiplicative approximation, but an additive one. Additive errors are invariant under scaling of experiments: that is, if each $\boldsymbol{v}_i$ is multiplied by a constant C (which one might do to force the log-determinant to be positive), an additive approximation does not change, whereas a multiplicative one does. In that respect, our results can be viewed as more robust.

Lastly, we discuss other recent variants of D-optimal design that are not directly related to our problem, but examine some related challenges. Mahabadi & Vuong (2022) studied group cardinality constraints as well as laminar matroid constraints. A more general matroid-constrained setting was considered by Madan et al. (2020) and Brown et al. (2022). Li et al. (2024) added an initial Fisher information matrix to the cardinality constrained problem, which essentially requires a specific subset of experiments to be included. Along similar lines, Ponte et al. (2023) added lower and upper bounds on the cardinality. These papers considered both exact branch-and-bound methods and approximation algorithms based on local search. Wang et al. (2023) combined the D-optimality criterion with vehicle routing constraints, motivated by a transportation context where data are collected by a vehicle. This paper proposed an exact approach using row and column generation. There has also been some work on D-optimal design with continuous decision variables: see, e.g., Vandenberghe et al. (1998) and Nikolov (2015). Closely related is the maximum entropy sampling problem (MESP), where similar exact and approximation methods have successfully been applied (Ko et al. 1995, Anstreicher 2020, Li & Xie 2023).

3 Local search with repetition (LS-W)

In this section, we introduce and analyze the LS-W algorithm. We begin with some notation that will be used in this and later sections. LS-W assumes that repetition is allowed, so we let x_i indicate the number of times that the ith experiment is conducted. Then, (1) can be recast as the

mixed-integer program

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \left[\det\left(\sum_{i\in[n]} x_i \boldsymbol{v}_i \boldsymbol{v}_i^\top\right)\right]^{\frac{1}{d}} \\ \text{s.t.} \quad & \sum_{i\in[n]} c_i x_i \le B \\ & x_i \in \mathbb{Z}^+, \forall i \in [n]. \end{aligned} \tag{MIP-W}$$

For any set S, we define $c(S) := \sum_{i\in S} c_i$ and $\boldsymbol{X}_S := \sum_{i\in S} \boldsymbol{v}_i \boldsymbol{v}_i^\top$. We also let $c_{\max} := \max_{i\in[n]} c_i$, and assume without loss of generality that $\min_{i\in[n]} c_i = 1$.

Our analysis proceeds in several steps. In Sections 3.1-3.2, we consider a simple case where each c_i is a non-negative power of $a = \frac{p}{q}$, where $p > q$ are positive coprime integers. For this setting, Section 3.1 states LS-W and shows its efficient implementation, while Section 3.2 proves the approximation guarantee by constructing a dual feasible solution based on the local optimality condition. Finally, in Section 3.3, we extend the result to general costs by rounding the costs $\boldsymbol{c}$ up to the closest powers of a.

Some additional notation will be useful. In Sections 3.1-3.2, we let $S_r := \{i \in S : c_i = a^r\}$ for any positive integer r. Our analysis of local optimality will use the quantities $\tau_{ij} := \boldsymbol{v}_i^\top \boldsymbol{X}_S^{-1} \boldsymbol{v}_j$ and $\tau_i := \boldsymbol{v}_i^\top \boldsymbol{X}_S^{-1} \boldsymbol{v}_i$, applied to sets S for which $\boldsymbol{X}_S^{-1}$ exists.

3.1 Local search algorithm

In cardinality-constrained D-optimal design (e.g., Madan et al. 2019), local search algorithms rely on simple interchanging rules: given a set S, we swap out some "selected" experiment $i \in S$ for some "unselected" experiment $j \notin S$ if doing so improves the objective function. This strategy cannot be directly applied when the costs are heterogeneous. For example, if $i \in S$ has a high cost, we may be able to achieve an improvement by removing it, but only if multiple $j \notin S$ with small costs are added. We develop a new notion of interchange that swaps, not two experiments, but two *packs* of experiments. For now, we consider a special case of the problem where all the experiments in each pack have the same cost, and the total costs of the two packs are also the same, thus ensuring that the interchange does not violate the budget constraint. Later, in Section 3.3, we will relax this requirement so that the costs can be "similar" without being identical.

The concept of interchanging packs presents a computational challenge, because the search space

of packs of experiments is much larger than that of individual experiments. We use the "strong superadditivity" property of the determinant to design a novel potential function that allows us to identify packs to interchange in a computationally efficient manner. Any interchange with a positive potential value guarantees an improvement in the objective function. On the other hand, nonpositive potential values provide a reasonable worst-case bound for the local search algorithm.

The full statement of LS-W is given in Algorithm 1. We begin with a detailed description of five aspects: 1) problem setting, 2) interchange rules, 3) initialization, 4) potential function, and 5) stopping criteria.

Problem setting. For now, we assume that $\{c_i\}_{i\in[n]} \subseteq \{1, a, ..., a^m\}$, where $a = \frac{p}{q}$. Let $M = \{k : |\{i \in [n] : c_i = a^k\}| \neq 0\} \subseteq \{0, 1, ..., m\}$ be the set of all possible exponents of the cost components. Without loss of generality, we assume that $0, m \in M$ (otherwise, we can either reduce m or rescale the costs and budget). Therefore, we have $c_{\max} := a^m$. We also assume that the budget satisfies $B \geq p^m d$, which, together with the fact that $c_{\max} \leq p^m$, ensures that every set of d experiments is feasible.

Interchange rules. Our theoretical results are proved for an "ordinary search" that identifies a promising interchange. We have also developed an alternate "subtle search" (line 19 in Algorithm 1) that performs better in practical implementation, but does not improve the theoretical guarantee. For this reason, the presentation of subtle search is relegated to Appendix A, and we focus on ordinary search in this discussion.

Ordinary search performs an interchange of two packs, of which one contains a number $p^{m-r}q^r$ of experiments with the same cost a^r, and the other contains $p^{m-\ell}q^\ell$ experiments with cost a^ℓ. Thus, both packs have total cost p^m. In each interchange, the experiments to be removed from S can be different, but the unselected experiments to be added into S must all be copies of the same experiment. In order to execute this procedure, S has to contain $p^{m-r}q^r$ experiments of cost a^r for some r, so it is necessary to find an initial feasible S satisfying this condition. However, once we do so, this property will persist throughout the ordinary search procedure.

Initialization. We begin by finding a set $\hat{S}$ of d distinct experiments for which $\boldsymbol{X}_{\hat{S}}$ is nonsingular. This can easily be done by taking the set of all experiments, and removing them one by one while maintaining nonsingularity; by assumption, it is always possible to do this.

Next, since the d experiments in $\hat{S}$ may not use up the entire budget, we augment them to ensure

that ordinary search can proceed. Since repetition is allowed, we may include $p^{m-r}q^r - 1$ additional copies of any $i \in \hat{S}$ that has cost a^r. Then, for any $r \in M$, the total number of experiments with cost a^r is either zero or $p^{m-r}q^r$. Letting I_0 denote the multi-set containing the indices of these experiments, we observe that the total cost of I_0 is $c(I_0) = p^m d$. If the budget permits, we may incorporate additional sets of experiments with total cost p^m.

When terminated, this procedure will return an initial design set I_0 satisfying the following conditions: 1) the matrix $\boldsymbol{X}_{I_0}$ is nonsingular; 2) the total cost satisfies $\lfloor \frac{B}{p^m} \rfloor p^m \leq c(I_0) \leq B$; and 3) $|\{i \in I_0 : c_i = c_{i_0}\}| \geq \frac{p^m}{c_{i_0}}$ for each $i \in I_0$. The first two conditions ensure that the initial set has a well-defined objective and its total cost is close to the budget. The third condition guarantees that ordinary search can be performed properly. It is worth mentioning that our approximation guarantees will hold for any other initialization strategy that satisfies these three conditions.

Potential function. The naive way to find the best interchange is to search over all possible sets of $p^{m-r}q^r$ experiments, for all $r \in M$, both inside and outside the incumbent set. Unfortunately, the number of possible subsets may be exponential. To expedite the process, we define the potential of two multisets I and J as

$$f(J, I) = \sum_{j \in J} \tau_j - \sum_{i \in I} \tau_i - \sum_{j \in J} \tau_j \sum_{i \in I} \tau_i + \sum_{j \in J} \sum_{i \in I} \tau_{ij}^2, \tag{2}$$

based on the strong superadditivity property of the determinant (Tie et al. 2011). We let I_r and J_ℓ denote the indices involved in a possible interchange of selected experiments with cost a^r and unselected experiments of cost a^ℓ. If $\max_{r,\ell} f(J_\ell, I_r) > 0$, we make the interchange that achieves the argmax.

It turns out that maximizing $f(J_\ell, I_r)$ over r, ℓ is much more efficient than searching over all possible subsets, and, in fact, can be done in polynomial time. This will be proved further down in Proposition 2. In brief, however, this maximization is not equivalent to exhaustive search: while any interchange satisfying $f(J_\ell, I_r) > 0$ will improve the objective function, the converse is not always true. However, restricting ourselves to interchanges with $f(J_\ell, I_r) > 0$ is sufficient for our theoretical guarantees.

Stopping criteria. The ordinary search terminates when $f(J_\ell, I_r) \leq 0$ for all $r, \ell \in M$. When this occurs, we arrive at line 18 in Algorithm 1, at which point the solution is considered to be locally

Algorithm 1 Local search algorithm for D-optimal design with repetition with costs a^i

1: Input: $B \geq p^m d$. Let I_0 be any multi-subset of $[n]$ such that $\boldsymbol{X}_{I_0} = \sum_{i \in I_0} \boldsymbol{v}_i \boldsymbol{v}_i^\top$ is non-singular, $c(I_0) \geq \lfloor \frac{B}{p^m} \rfloor p^m$ and $|\{i \in I_0 : c_i = c_{i_0}\}| \geq \frac{p^m}{c_{i_0}}$ for each $i_0 \in I_0$. $\tilde{S} = I_0$. $\boldsymbol{X}_{\tilde{S}} = \sum_{i \in I_0} \boldsymbol{v}_i \boldsymbol{v}_i^\top$. Compute $\boldsymbol{\Lambda} := \boldsymbol{X}_{\tilde{S}}^{-1}$, $\tau_{ij} := \boldsymbol{v}_i^\top \boldsymbol{\Lambda} \boldsymbol{v}_j$ for all $i \in \tilde{S}$, $j \in [n]$, and $\tau_j := \boldsymbol{v}_j^\top \boldsymbol{\Lambda} \boldsymbol{v}_j$ for all $j \in [n]$.
2: **for** $r, \ell \in M$ **do**
3: $\quad \tilde{S}_r = \{i \in \tilde{S} : c_i = a^r\}$
4: $\quad$ **if** $\tilde{S}_r \neq \emptyset$ **then**
5: $\quad\quad J_\ell, I_r = \underset{J,I}{\operatorname{argmax}} \{f(J, I) : |I| = p^{m-r} q^r, I \subseteq \tilde{S}_r, c_j = a^\ell, J = \{ \underbrace{j, \cdots, j}_{p^{m-\ell} q^\ell \text{ copies}} \}\}$.
6: $\quad$ **end if**
7: **end for**.
8: $r_0, \ell_0 = \operatorname{argmax}_{r,\ell} f(J_\ell, I_r)$.
9: **if** $f(J_{\ell_0}, I_{r_0}) > 0$ **then**
10: $\quad$ Add the multi-set J_{ℓ_0} to $\tilde{S}$ and delete the multi-set I_{r_0} from $\tilde{S}$.
11: $\quad \boldsymbol{X}_{\tilde{S}} = \boldsymbol{X}_{\tilde{S}} + \sum_{j \in J_{\ell_0}} \boldsymbol{v}_j \boldsymbol{v}_j^\top - \sum_{i \in I_{r_0}} \boldsymbol{v}_i \boldsymbol{v}_i^\top$.
12: $\quad \boldsymbol{\Lambda} = \boldsymbol{\Lambda} - \frac{\sum_{j \in J_{\ell_0}} \boldsymbol{\Lambda} \boldsymbol{v}_j \boldsymbol{v}_j^\top \boldsymbol{\Lambda}}{1 + \sum_{j \in J_{\ell_0}} \tau_j}$
13: $\quad$ **for** $i \in I_{r_0}$ **do**
14: $\quad\quad \boldsymbol{\Lambda} = \boldsymbol{\Lambda} + \frac{\boldsymbol{\Lambda} \boldsymbol{v}_i \boldsymbol{v}_i^\top \boldsymbol{\Lambda}}{1 - \boldsymbol{v}_i^\top \boldsymbol{\Lambda} \boldsymbol{v}_i}$
15: $\quad$ **end for**
16: $\quad$ Update $\tau_{ij} = \boldsymbol{v}_i^\top \boldsymbol{\Lambda} \boldsymbol{v}_j$ for all $i \in \tilde{S}$, $j \in [n]$, and $\tau_j = \boldsymbol{v}_j^\top \boldsymbol{\Lambda} \boldsymbol{v}_j$ for all $j \in [n]$.
17: $\quad$ Return to line 2.
18: **end if**.
19: $\tilde{S}, \boldsymbol{X}_{\tilde{S}} = SubtleSearchW(\tilde{S}, \boldsymbol{X}_{\tilde{S}}, \boldsymbol{\Lambda})$. ▷ An optional step to improve the objective.
20: **return** $\tilde{S}, \boldsymbol{X}_{\tilde{S}}$.

optimal. This is the solution for which we prove the approximation guarantee.

We now study the properties of Algorithm 1. Our first task is to show that it will terminate in a finite number of iterations. We state two technical lemmas, established in prior work, that are key to this result; since our approach heavily relies on the superadditivity of the determinant, it is worth stating this property for the reader's reference. We then state the result itself (Proposition 1), deferring the proof to the Appendix due to space considerations.

Lemma 1 (Strong superadditivity; Lemma 2.2 in Tie et al. 2011). For any $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \succeq 0$,

$$\det(\boldsymbol{A} + \boldsymbol{B} + \boldsymbol{C}) + \det(\boldsymbol{C}) \geq \det(\boldsymbol{A} + \boldsymbol{C}) + \det(\boldsymbol{B} + \boldsymbol{C}).$$

Lemma 2 (Lemmas 8 and 10 in Madan et al. 2019). Suppose $\boldsymbol{X}_S = \sum_{i \in S} \boldsymbol{v}_i \boldsymbol{v}_i^\top$ is an invertible matrix, $\tau_i := \boldsymbol{v}_i^\top \boldsymbol{X}_S^{-1} \boldsymbol{v}_i$, and $\tau_{ij} := \boldsymbol{v}_i^\top \boldsymbol{X}_S^{-1} \boldsymbol{v}_j$. Then:

(i) For any $j \in [n]$,

$$\det\left(\boldsymbol{X}_S + \boldsymbol{v}_j \boldsymbol{v}_j^\top\right) = \det\left(\boldsymbol{X}_S\right)\left(1 + \tau_j\right).$$

(ii) For any $i \in S$,

$$\det\left(\boldsymbol{X}_S - \boldsymbol{v}_i \boldsymbol{v}_i^\top\right) = \det\left(\boldsymbol{X}_S\right)\left(1 - \tau_i\right).$$

(iii) For any $i \in S$, $j \in [n]$,

$$\det\left(\boldsymbol{X}_S + \boldsymbol{v}_j \boldsymbol{v}_j^\top - \boldsymbol{v}_i \boldsymbol{v}_i^\top\right) = \det\left(\boldsymbol{X}_S\right)\left(1 + \tau_j - \tau_i - \tau_j \tau_i + \tau_{ij}^2\right).$$

(iv) The following identity must hold:

$$\sum_{i \in S} \tau_i = d.$$

(v) For any $j \in [n]$,

$$\sum_{i \in S} \tau_{ij}^2 = \tau_j.$$

Proposition 1. In each iteration of Algorithm 1, the following statements hold:

(i) the determinant $\det(\boldsymbol{X}_{\tilde{S}})$ is monotonically strictly increasing from the previous iteration;

(ii) the inverse $\boldsymbol{X}_{\tilde{S}}^{-1}$ is always well-defined;

(iii) the algorithm always terminates in a finite number of iterations.

As mentioned previously, it is possible to have a feasible interchange that improves the objective value $\det(\boldsymbol{X}_{\tilde{S}})$, yet has negative potential. Essentially, the condition $f(J_\ell, I_r) > 0$ is stronger than simply ensuring a positive increment of the determinant. In the following example, we present one instance where this occurs.

Example 1. Consider a set of 7 potential experiments of dimension $d = 2$, where $\boldsymbol{v}_1 = (1, 0)^\top$, $\boldsymbol{v}_2 = (0, 1)^\top$, $\boldsymbol{v}_3 = \boldsymbol{v}_4 = (\frac{1}{\sqrt{2}}, 0)^\top$, $\boldsymbol{v}_5 = \boldsymbol{v}_6 = (0, \frac{1}{\sqrt{2}})^\top$ and $\boldsymbol{v}_7 = (2, 0)^\top$. Let the cost components $\boldsymbol{c}$ be $c_1 = c_2 = 2$, $c_3 = c_4 = c_5 = c_6 = 1$ and $c_7 = 4$. Let $B = 8$ be the budget. When implementing Algorithm 1, let us use $p = 2, q = 1, m = 2$ with initial design set $\tilde{S} = \{1, 2, 3, 4, 5, 6\}$. In this case, we begin with the matrix $\boldsymbol{X}_{\tilde{S}} = \sum_{i=1}^{6} \boldsymbol{v}_i \boldsymbol{v}_i^\top = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$. According to the interchange rule, our

choices are to replace either (i) $\boldsymbol{v}_1, \boldsymbol{v}_2$ or (ii) $\boldsymbol{v}_3, \boldsymbol{v}_4, \boldsymbol{v}_5, \boldsymbol{v}_6$ by either (i) 2 copies of $\boldsymbol{v}_1$ or $\boldsymbol{v}_2$, or (ii) 4 copies of $\boldsymbol{v}_3$ or $\boldsymbol{v}_4$ or $\boldsymbol{v}_5$ or $\boldsymbol{v}_6$, or (iii) 1 copy of $\boldsymbol{v}_7$. By direct computation, we can show that the objective function is improved by only two of these interchanges: (i) replace $\boldsymbol{v}_1, \boldsymbol{v}_2$ by $\boldsymbol{v}_7$, or (ii) replace $\boldsymbol{v}_3, \boldsymbol{v}_4, \boldsymbol{v}_5, \boldsymbol{v}_6$ by $\boldsymbol{v}_7$. However, neither of them is accepted by Algorithm 1. This is because the induced matrix corresponding to either of the aforementioned interchanges is

$$\boldsymbol{Y} := \boldsymbol{X} - \sum_{i=1}^{2} \boldsymbol{v}_i \boldsymbol{v}_i^\top + \boldsymbol{v}_7 \boldsymbol{v}_7^\top = \boldsymbol{X} - \sum_{i=3}^{6} \boldsymbol{v}_i \boldsymbol{v}_i^\top + \boldsymbol{v}_7 \boldsymbol{v}_7^\top = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}.$$

Clearly, $\det(\boldsymbol{X}) = 4 < 5 = \det(\boldsymbol{Y})$. On the other hand, since $\boldsymbol{X}^{-1} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}$, we have

$$\tau_1 = (1, 0) \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2}.$$

Similarly, we can compute $\tau_2 = \frac{1}{2}$, $\tau_3 = \tau_4 = \tau_5 = \tau_6 = \frac{1}{4}$, $\tau_7 = 2$, $\tau_{1,7} = 1$, $\tau_{3,7} = \tau_{4,7} = \frac{1}{\sqrt{2}}$, $\tau_{2,7} = \tau_{5,7} = \tau_{6,7} = 0$. However, the potential function values for both interchanges are:

$$f(\{7\}, \{1, 2\}) = \tau_7 - \left(\sum_{i=1}^{2} \tau_i\right) - \tau_7 \left(\sum_{i=1}^{2} \tau_i\right) + \left(\sum_{i=1}^{2} \tau_{i,7}^2\right) = 0,$$
$$f(\{7\}, \{3, 4, 5, 6\}) = \tau_7 - \left(\sum_{i=3}^{6} \tau_i\right) - \tau_7 \left(\sum_{i=3}^{6} \tau_i\right) + \left(\sum_{i=3}^{6} \tau_{i,7}^2\right) = 0.$$

Therefore, Algorithm 1 will stop with the initial design. □

The preceding example clearly shows how Algorithm 1 can overlook opportunities to improve the objective. On the other hand, our interchange rule is far more efficient than exhaustive search. The following result establishes polynomial-time complexity.

Proposition 2. The time complexity of each interchange of the ordinary search (i.e., running lines 2-18 once) in Algorithm 1 is $O(Bn(d^2 + m\log(p)))$.

We can use the result of Proposition 2 to quantify the time complexity of Algorithm 1. Let L be the maximum binary encoding length of the vectors $\boldsymbol{v}_i$ over $i \in [n]$. Given an optimal (multi-) subset S^* for problem (1), it can be shown (using similar arguments as in Madan et al. (2019)) that

the ratio between the optimal objective value $\det(\sum_{i\in S^*} \boldsymbol{v}_i \boldsymbol{v}_i^\top)$ and the determinant value of the non-singular initialization $\det(\sum_{i\in I_0} \boldsymbol{v}_i \boldsymbol{v}_i^\top)$ is at most $B^d 2^{8Ld^2}$. Thus, if each interchange increases the objective value by a multiplicative factor $(1+\delta)$, then the number of interchanges is at most $O(\frac{Ld^3 \log(B)}{\delta})$. Combining this with Proposition 2, we find that the overall running time of LS-W is $O(\delta^{-1} Lnd^3(d^2 + m\log(p))B\log(B))$, as stated in Table 1.

3.2 Local optimality and approximation guarantee

Let $\tilde{S}$ be the index set of the final solution obtained when Algorithm 1 terminates. Our main result, stated in Theorem 1, derives an approximation guarantee for $\tilde{S}$ by constructing a feasible solution to the dual of the continuous relaxation of (MIP-W). At a high level, this technique is similar to the one used by Madan et al. (2019) for cardinality-constrained D-optimal design. The crucial difference between the two proofs lies in a certain bound used to show dual-feasibility. In our setting, we require a uniform upper bound on the ratio $\frac{\tau_j}{c_j}$ (which can be viewed as the marginal improvement per unit cost achievable by adding another copy of experiment j) for all $j \in [n]$ and index set $\tilde{S}$. The following lemma states the bound. The proof is technical and is relegated to Appendix B.1.

Lemma 3. After line 18 of Algorithm 1, we have

$$\frac{\tau_j}{c_j} \leq \frac{d}{c(\tilde{S}) - p^m d + p^m}, \quad \forall j \in [n].$$

Now, we are ready to prove the approximation guarantee. The proof proceeds by relaxing the original problem, computing its dual, and using $\boldsymbol{X}_{\tilde{S}}$ to construct a dual-feasible solution; the guarantee then follows by weak duality. This structure is overall similar to the proof of Theorem 7 in Madan et al. (2019), with Lemma 3 being the key ingredient. The proof is relegated to Appendix B.1.

Theorem 1. Let $\tilde{S}$ be the output of Algorithm 1. Then we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{\left(\frac{B}{p^m} - \left\lfloor \frac{B}{p^m} \right\rfloor + d - 1\right) p^m}{B}\right) \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{p^m d}{B}\right).$$

Particularly, when $B \geq \frac{p^m d}{\epsilon}$ for some $\epsilon \in (0,1)$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq (1-\epsilon)\left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

The following corollary observes that, if $c_i \equiv 1$ (i.e., the problem reduces the cardinality-constrained case), Theorem 1 recovers the strongest existing bound in the literature Madan et al. (2019).

Corollary 1. When the budget $B = k$, each cost component $c_i = 1$ for all $i \in [n]$. Let us choose $p = q = 1$ and let $\tilde{S}$ be the output of Algorithm 1. Then we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}\left(1 - \frac{d-1}{k}\right).$$

Particularly, when $k \geq \frac{d-1}{\epsilon}$ for some $\epsilon \in (0,1)$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq (1-\epsilon)\left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

3.3 Approximation guarantee for general costs

Thus far, we have assumed that each c_i is equal to some positive power of $a = \frac{p}{q}$. To handle general costs, the main idea is to round up each c_i to the nearest power of a_i. In other words, we calculate $c'_i := a^{\lceil \log_a(c_i) \rceil}$. Note that we may choose p, q at our convenience to mitigate rounding error. We then apply Algorithm 1 using the inputs $B, \boldsymbol{c}', \{\boldsymbol{v}_i\}_{i\in[n]}, p, q$ and $m := \lceil \log_a(c_{\max}) \rceil$. Since the cost components of $\boldsymbol{c}$ are rounded up to $\boldsymbol{c}'$, it is guaranteed that the set $\tilde{S}$ returned by Algorithm 1 is feasible with respect to the original costs.

Under the modified inputs, Lemma 3 yields

$$\frac{\tau_j}{c'_j} \leq \frac{d}{\sum_{j\in\tilde{S}} c'_j - p^m d + p^m} \quad \forall j \in [n].$$

However, since $\frac{c'_i}{c_i} < a$ for any $i \in [n]$, we have

$$\frac{\tau_j}{c_j} \leq \frac{c'_j}{c_j}\frac{d}{\sum_{j\in\tilde{S}} c'_j - p^m d + 1} < \frac{ad}{\sum_{j\in\tilde{S}} c'_j - p^m d + p^m} \quad \forall j \in [n],$$

thus increasing the right-hand side of the bound by at most a factor of a.

Then, by modifying the proof of Theorem 1, we obtain the following result. The detailed proof can be found in Appendix B.1.

Theorem 2. Let $\tilde{S}$ be the output of Algorithm 1 with $a = \frac{p}{q}$, $c'_i := a^{\lceil \log_a(c_i) \rceil}$, and $m = \lceil \log_a(c_{\max}) \rceil$. When $B \geq p^m d$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(\frac{q}{p} - q^m \frac{c_{\max} d}{B}\right).$$

Particularly, when $B \geq q^m \frac{c_{\max} d}{\epsilon}$ for some $\epsilon \in (0, \frac{q}{p})$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left(\frac{q}{p} - \epsilon\right) \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

We note that the choice of parameters p, q affects the approximation guarantee in two ways. First, it affects the approximation ratio through the rounding error $\frac{c_i}{c'_i}$. The approximation ratio can be strengthened to a $(\min_{i \in [n]} \frac{c_i}{c'_i} - \epsilon)$-approximation if $\frac{c_i}{c'_i}$ is tighter. At the same time, it affects the threshold required by our approximation guarantee. Between these two considerations, we recommend prioritizing the reduction of rounding error.

4 Local search without repetition (LS-WO)

In this section, repetition is no longer allowed, and the mixed-integer formulation (MIP-W) is replaced by

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \left[\det\left(\sum_{i \in [n]} x_i \boldsymbol{v}_i \boldsymbol{v}_i^\top\right)\right]^{\frac{1}{d}} \\ \text{s.t.} \quad & \sum_{i \in [n]} c_i x_i \leq B \\ & x_i \in \{0, 1\}, \forall i \in [n]. \end{aligned} \tag{MIP-WO}$$

LW-WO uses the same basic idea as LS-W. We first assume that all costs are powers of a, and iteratively interchange two packs of experiments with the same total cost; we then handle general costs using rounding. There are, however, several nontrivial difficulties in extending the analysis of Section 3:

- When making an interchange, Algorithm 1 always adds multiple copies of the same experiment.

This is no longer possible when repetition is not allowed. In this setting, we must consider sets of distinct experiments, which is more difficult to do efficiently.

- It may happen that we do not have enough unselected experiments to create pack with the desired total cost. We resolve this issue by creating artificial experiments with data vectors $\boldsymbol{v} = 0$ and costs a^ℓ as needed.

- If repetition is not allowed, the bound in Lemma 3 holds only for $j \notin \tilde{S}$, not for all $j \in [n]$, complicating the proof of the approximation ratio.

- The continuous relaxation of the problem without repetition has more dual variables, making it more challenging to construct a dual-feasible solution.

For these reasons, we replace Algorithm 1 by Algorithm 2, with several main differences that are discussed in Section 4.1 for costs that are powers of a. General costs are covered in Section 4.2.

4.1 Costs that are powers of a

Again, we suppose that, for all $i \in [n]$, we have $c_i = a^\ell$ for some $\ell \in M$. While Algorithm 2 is similar to Algorithm 1, they differ in four aspects:

Problem setting. For each $\ell \in M$, we add $\lfloor \frac{B}{p^m} \rfloor p^{m-\ell} q^\ell - 1$ artificial experiments with zero value and cost a^ℓ.

Interchange rules. We still interchange packs of experiments whose total cost is p^m, but we also allow the interchange of two individual experiments that have the same cost. Of course, we can no longer add multiple copies of the same experiment.

Initialization. In the same way as before, we construct an initial set $\hat{S}$ of d distinct experiments whose induced matrix $\boldsymbol{X}_{\hat{S}}$ is nonsingular. However, this set is now augmented by adding artificial experiments with zero value, rather than making copies of the initial d experiments. At the end of this procedure, we have a set $I_0 \subseteq [n]$ such that $\sum_{i \in I_0} v_i v_i^\top$ is nonsingular, $c(I_0) = \lfloor \frac{B}{p^m} \rfloor p^m$, and $|\{i \in I_0 : c_i = c_{i_0}\}| \geq \frac{p^m}{c_{i_0}}$ for all $i_0 \in I_0$.

Potential function. The stopping criterion is the same as in Algorithm 1, but the potential function (line 6 of Algorithm 2) is different. The new function is still based on superadditivity, but is motivated by the following inequality, whose detailed proof can be found in Appendix B.2.

Lemma 4. For any two index subsequences with distinct elements $I := \{i_1, i_2, \ldots, i_{p^{m-r}q^r}\} \subseteq \tilde{S}$ and $J := \{j_1, j_2, \ldots, j_{p^{m-r}q^r}\} \subseteq \tilde{S}^C := [n] \setminus \tilde{S}$, we have

$$\det\left(\boldsymbol{X}_{\tilde{S}} + \sum_{k=1}^{p^{m-\ell}q^{\ell}} \boldsymbol{v}_{j_k}\boldsymbol{v}_{j_k}^{\top} - \sum_{k=1}^{p^{m-r}q^{r}} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^{\top}\right) \geq \det\left(\boldsymbol{X}_{\tilde{S}}\right)\left(1 + \sum_{k=1}^{p^{m-\ell}q^{\ell}} \tau_{j_k}\right)\left(1 - \sum_{k=1}^{p^{m-r}q^{r}} \tau_{i_k}\right). \quad (3)$$

The right-hand side of (3) is a separable product of one term depending only on the unselected experiments j_k to be "added" and one term depending only on the selected experiments i_k to be "deleted." Formally, the potential function for deleting experiments indexed by I_m and adding experiments indexed by J_m is defined as

$$f(J_m, I_m) := \sum_{j \in J_m} \tau_j - \sum_{i \in I_m} \tau_i - \sum_{i \in I_m} \tau_i \sum_{j \in J_m} \tau_j. \quad (4)$$

It is fairly straightforward to extend several proofs in Section 3.1 to obtain the following results. Therefore, we state them without proof. It is worth noting, however, that the interchange still has polynomial-time complexity, despite the fact that we can no longer add multiple copies of the same experiment. This happens because the new potential function (4) no longer contains the cross terms τ_{ij}, so we may compute τ_i and τ_j separately for added and deleted experiments.

Proposition 3. In each iteration of Algorithm 2, the following statements hold:

(i) the determinant $\det(\boldsymbol{X}_{\tilde{S}})$ is monotonically strictly increasing from the previous iteration;

(ii) the inverse $\boldsymbol{X}_{\tilde{S}}^{-1}$ is always well-defined;

(iii) the algorithm always terminates in a finite number of iterations.

Furthermore, the time complexity of each interchange of the ordinary search (i.e., running lines 4-22 once) in Algorithm 2 is $O(n(d^2 + m\log(p)))$.

It is worth noting that, by combining Proposition 3 with the arguments at the end of Section 3.1, we obtain an overall time complexity of $O(\delta^{-1}Lnd^3(d^2 + m\log(p))\log(B))$ for LS-WO, with δ being the minimum multiplicative factor by which the objective value improves in one iteration. This is the quantity reported in Table 1. It can be readily seen that LS-WO is more efficient than LS-W, as it incurs a smaller cost per interchange.

Algorithm 2 Local search algorithm for D-optimal design without repetition with costs a^i

1: Input: $B \geq p^m(d+m+1)$. Let I_0 be any subset of $[n]$ such that $\boldsymbol{X}_{I_0} = \sum_{i\in I_0} \boldsymbol{v}_i\boldsymbol{v}_i^\top$ is non-singular, $c(I_0) \geq \lfloor \frac{B}{p^m} \rfloor p^m$ and $|\{i \in I_0 : c_i = c_{i_0}\}| \geq \frac{p^m}{c_{i_0}}$ for each $i_0 \in I_0$. $\tilde{S} = I_0$. $\boldsymbol{X}_{\tilde{S}} = \sum_{i\in I_0} \boldsymbol{v}_i\boldsymbol{v}_i^\top$. Compute $\boldsymbol{\Lambda} := \boldsymbol{X}_{\tilde{S}}^{-1}$, $\tau_j := \boldsymbol{v}_j^\top \boldsymbol{\Lambda} \boldsymbol{v}_j$ for all $j \in [n]$.
2: $\tilde{S}_r = \{i \in S : c_i = a^r\}$.
3: $\tilde{S}_\ell^C = \{j \notin S : c_j = a^\ell\}$.
4: $I_m = \operatorname{argmin}_{I:r, I\subseteq \tilde{S}_r, |I| = p^{m-r}q^r} \sum_{i\in I} \tau_i$. ▷ Only r with $\tilde{S}_r \neq \emptyset$ is considered.
5: $J_m = \operatorname{argmax}_{J:\ell, J\subseteq \tilde{S}_\ell^C, |J| = p^{m-\ell}q^\ell} \sum_{j\in J} \tau_j$. ▷ Only ℓ with $\tilde{S}_\ell^C \neq \emptyset$ is considered.
6: $f(J_m, I_m) = \sum_{j\in J_m} \tau_j - \sum_{i\in I_m} \tau_i - \sum_{i\in I_m} \tau_i \sum_{j\in J_m} \tau_j$
7: **if** $f(J_m, I_m) \leq 0$ **then**
8: $\quad J_m, I_m = \operatorname{argmax}_{J,I:J=\{j\},I=\{i\},c_i=c_j} (\tau_j - \tau_i + \tau_i\tau_j)$.
9: $\quad$ **if** $f(J_m, I_m) \leq 0$ **then**
10: $\quad\quad$ Go to line 23.
11: $\quad$ **end if**
12: **end if**.
13: $\tilde{S} = \tilde{S} \cup J_m \backslash I_m$.
14: $\boldsymbol{X}_{\tilde{S}} = \boldsymbol{X}_{\tilde{S}} + \sum_{j\in J_m} \boldsymbol{v}_j\boldsymbol{v}_j^\top - \sum_{i\in I_m} \boldsymbol{v}_i\boldsymbol{v}_i^\top$.
15: **for** $j \in J_m$ **do**
16: $\quad \boldsymbol{\Lambda} = \boldsymbol{\Lambda} - \frac{\boldsymbol{\Lambda}\boldsymbol{v}_j\boldsymbol{v}_j^\top\boldsymbol{\Lambda}}{1+\boldsymbol{v}_j^\top\boldsymbol{\Lambda}\boldsymbol{v}_j}$
17: **end for**
18: **for** $i \in I_m$ **do**
19: $\quad \boldsymbol{\Lambda} = \boldsymbol{\Lambda} + \frac{\boldsymbol{\Lambda}\boldsymbol{v}_i\boldsymbol{v}_i^\top\boldsymbol{\Lambda}}{1-\boldsymbol{v}_i^\top\boldsymbol{\Lambda}\boldsymbol{v}_i}$
20: **end for**
21: Update $\tau_j = \boldsymbol{v}_j^\top\boldsymbol{\Lambda}\boldsymbol{v}_j$ for all $j \in [n]$.
22: Return to line 2.
23: $\tilde{S}, \boldsymbol{X}_{\tilde{S}} = SubtleSearchWO(\tilde{S}, \boldsymbol{X}_{\tilde{S}}, \boldsymbol{\Lambda})$. ▷ An optional step to improve the objective.
24: **return** $\tilde{S}, \boldsymbol{X}_{\tilde{S}}$.

The main obstacle remaining in the way of the approximation guarantee is that Lemma 3 no longer holds in our setting. We derive new bounds based on the local optimality condition

$$\sum_{j\in J_m} \tau_j - \sum_{i\in I_m} \tau_i - \sum_{i\in I_m} \tau_i \sum_{j\in J_m} \tau_j \leq 0, \tag{5}$$

which hold for the set $\tilde{S}$ returned by Algorithm 2 when ordinary search terminates (i.e., when we reach line 23). Define $\tilde{S}_r := \{i \in \tilde{S} : c_i = a^r\}$, $\tilde{S}_\ell^C := \{j \notin \tilde{S} : c_j = a^\ell\}$ for each $r, \ell \in M$. Then, we may further rewrite (5) for each type of interchange: (i) interchanges of two packs of total cost p^m, lines 2-6; and (ii) interchanges of individual experiments with equal cost, lines 7-12).

Interchanges of packs. For any index sequence $I_m = \{i_k\}_{k\in[p^{m-r}q^r]} \subseteq \tilde{S}_r$ with $r \in M$, and any $J_m = \{j_k\}_{k\in[p^{m-\ell}q^\ell]} \subseteq \tilde{S}_\ell^C$ with $\ell \in M$, the local optimality condition (5) reduces to

$$\frac{\sum_{k=1}^{p^{m-\ell}q^\ell} \tau_{j_k}}{p^m}\left(1 - p^m \frac{\sum_{i=1}^{p^{m-r}q^r} \tau_{i_k}}{p^m}\right) - \frac{\sum_{k=1}^{p^{m-r}q^r} \tau_{i_k}}{p^m} \leq 0. \tag{6}$$

Define $\omega := \frac{1}{p^m} \min_{r, I\subseteq\tilde{S}_r, |I|=p^{m-r}q^r} \sum_{i\in I} \tau_i$. The following result for ω aids in rewriting (6) and will be used to prove the approximation guarantee.

Proposition 4. The following inequalities must hold: (i) $\omega \leq \frac{d}{c(\tilde{S})}$, and (ii) $1 - p^m\omega > 0$.

By Proposition 4, maximizing the left-hand side of (6) results in an equivalent optimality condition

$$\frac{\sum_{k=1}^{p^{m-\ell}q^\ell} \tau_{j_k}}{p^m}(1 - p^m\omega) - \omega \leq 0. \tag{7}$$

Since $c_j = a^\ell$ for any $j \in \tilde{S}_\ell^C$, (7) can be rewritten as

$$\frac{1}{p^{m-\ell}q^\ell} \sum_{k=1}^{p^{m-\ell}q^\ell} \frac{\tau_{j_k}}{c_{j_k}} \leq \frac{\omega}{1 - p^m\omega}. \tag{8}$$

Thus, we no longer have a uniform bound on the ratios $\frac{\tau_j}{c_j}$, as we did in Lemma 3. Instead, (8) is a bound on an *average* of these ratios. Some of the individual ratios may therefore violate the bound. Let us define "outlier" sets

$$O_\ell^J = \left\{j \in \tilde{S}_\ell^C : \frac{\tau_j}{c_j} > \frac{\omega}{1 - p^m\omega}\right\}, \quad O_r^I = \left\{i \in \tilde{S}_r : \frac{\tau_i}{c_i} < \omega\right\}.$$

of experiments whose $\frac{\tau}{c}$ ratios behave differently from the average. To control the number of such outliers, we require an additional interchange rule for individual experiments.

Interchanges of individual experiments. In this case, $I_m = \{i\}$ and $J_m = \{j\}$. In other words, the optimality condition (5) for this type of interchange becomes

$$\tau_j(1 - \tau_i) - \tau_i \leq 0, \tag{9}$$

for any $\ell \in M$, $j \in \tilde{S}_\ell^C$, and $i \in \tilde{S}_\ell$. The following result proves that, under (9), the number of outliers becomes very small; the proofs can be found in Appendix B.2.

Proposition 5. For each $\ell \in M$, the following statements hold: (i) $|O_\ell^J| \leq p^{m-\ell}q^\ell - 1$ and $|O_\ell^I| \leq p^{m-\ell}q^\ell - 1$; and (ii) $|O_\ell^J| \times |O_\ell^I| = 0$.

Together, these bounds substitute for Lemma 3 and enable us to prove the approximation guarantee. As in the proof of Theorem 1, we formulate the dual of the continuous relaxation of (MIP-WO), construct a dual feasible solution, and use weak duality to bound the objective value of $\tilde{S}$. One noteworthy difference is that the bound obtained from weak duality now includes some additional terms involving the outlier sets, and Proposition 5 is needed to bound these terms. The full proof of the theorem is given in Appendix B.2

Theorem 3. Let $\tilde{S}$ be the output of Algorithm 2. Then we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{(d+m+1)\,p^m - \sum_{r=0}^{m-1} a^r}{B + \left(mp^m - \sum_{r=0}^{m-1} a^r\right)}\right).$$

Particularly, when $B \geq \epsilon^{-1}((d+m+1)p^m - \sum_{r=0}^{m-1} a^r) - (mp^m - \sum_{r=0}^{m-1} a^r)$ for some $\epsilon \in (0,1)$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq (1-\epsilon)\left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}$$

Once more, in the special case of equal costs, the problem reduces to cardinality-constrained D-optimal design (without repetition), and Theorem 3 recovers the strongest existing bound for that problem (Madan et al. 2019).

Corollary 2. Suppose that $B = k$ and $c_i \equiv 1$. Choose $p = q = 1$ and let $\tilde{S}$ be the output of Algorithm 2. Then, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{d}{k}\right).$$

Particularly, when $k \geq \frac{d}{\epsilon}$ for some $\epsilon \in (0,1)$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq (1-\epsilon)\left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

4.2 Approximation guarantee for general costs

As in Section 3.3, let $c'_i := (\frac{p}{q})^{\lceil \log_{\frac{p}{q}}(c_i) \rceil}$ be the rounded cost of each $i \in [n]$. Next, we apply Algorithm 2 with the inputs $B, \boldsymbol{c}', \{\boldsymbol{v}_i\}_{i\in[n]}, p, q$ and $m := \lceil \log_a(c_{\max}) \rceil$. Using a somewhat more complicated version of the techniques discussed in Section 3.3, we obtain an approximation guarantee for the general case. The detailed proof can be found in Appendix B.2.

Theorem 4. Let $\tilde{S}$ be the output of Algorithm 2 with inputs $a = \frac{p}{q}$, $c'_i := a^{\lceil \log_a(c_i) \rceil}$, $m = \lceil \log_a(c_{\max}) \rceil$. If $B \geq p^m(d+m+1)$, then we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(\frac{q}{p} - q^m \frac{c_{\max}\,(d+m+1)}{B}\right).$$

Particularly, when $B \geq q^m \frac{c_{\max}(d+m+1)}{\epsilon}$ for some $\epsilon \in (0, \frac{q}{p})$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left(\frac{q}{p} - \epsilon\right) \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

5 Sampling with repetition (SA-W)

Sampling algorithms for budget-constrained D-optimal design are, overall, easier to construct than local search methods, and can rely more on existing results. In this and the next section, we streamline our presentation and focus on the new details required to complete the extension.

As in Sections 3.3 and 4.2, each cost c_i is rounded up to the nearest power of a. For each $r \in M$, we then pack $p^{m-r}q^r$ copies of each experiment whose rounded cost is a^r. More precisely, for every vector $\boldsymbol{v}_i$ for which $c'_i = a^r$, we create an aggregated vector $\boldsymbol{u}_i$ such that $\boldsymbol{u}_i\boldsymbol{u}_i^\top = p^{m-r}q^r\boldsymbol{v}_i\boldsymbol{v}_i^\top$. Then, every pack has the same total cost p^m, which essentially reduces the problem to an instance of cardinality-constrained D-optimal design with repetition.

For notational convenience, let us denote $k_i := \lceil \log_{\frac{p}{q}} c_i \rceil$ for each $i \in [n]$, $m := \max_i k_i$. We take

the logarithm of the objective function of (MIP-W) and relax the problem as

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \frac{1}{d} \log\det\left(\sum_{i\in[n]} x_i \boldsymbol{v}_i \boldsymbol{v}_i^\top\right) \\ \text{s.t.} \quad & \sum_{i\in[n]} c_i x_i \le B \\ & x_i \ge 0 \quad \forall i \in [n]. \end{aligned} \tag{REL-W}$$

Let $w_1(B)$ denote the optimal objective value of the relaxation (REL-W). We also let $\tilde{c}_i = \frac{c_i}{a^{k_i}}$ for each $i \in [n]$, and define $B' = \frac{B}{p^m}$. Then (REL-W) is equivalent to the program

$$\begin{aligned} w_2\left(B'\right) = \max_{\boldsymbol{y}} \quad & \frac{1}{d} \log\det\left(\sum_{i\in[n]} y_i \boldsymbol{u}_i \boldsymbol{u}_i^\top\right) \\ \text{s.t.} \quad & \sum_{i\in[n]} \tilde{c}_i y_i \le B' \\ & y_i \ge 0 \quad i \in [n]. \end{aligned} \tag{REL-W-2}$$

The transformation $x_i = p^{m-k_i} q_i^k y_i$ can be used to convert a feasible solution of (REL-W) to one of (REL-W-2) and vice versa. It can readily be seen that

$$w_2\left(\frac{B}{p^m}\right) = w_1(B). \tag{10}$$

The costs $\tilde{c}_i$ satisfy $\frac{q}{p} < \tilde{c}_i \le 1$. We may view them as being "close to 1," motivating an approximate problem in which $\tilde{c}_i$ in (REL-W-2) is replaced by 1, and the right-hand side is rounded down to the interest integer. This problem is an instance of cardinality-constrained D-optimal design (with cardinality $B'' = \lfloor B' \rfloor$):

$$\begin{aligned} w_3\left(B''\right) = \max_{\boldsymbol{y}} \quad & \frac{1}{d} \log\det\left(\sum_{i\in[n]} y_i \boldsymbol{u}_i \boldsymbol{u}_i^\top\right) \\ \text{s.t.} \quad & \sum_{i\in[n]} y_i \le B'' \\ & y_i \ge 0 \quad i \in [n]. \end{aligned} \tag{REL-W-3}$$

We may then run Algorithm 3, which applies the techniques of Singh & Xie (2020) to (REL-W-3).

Algorithm 3 Sampling algorithm for budget-constrained D-optimal design with repetition.

Initialize multi-set $\tilde{S} = \emptyset$ and number $\epsilon \in (0,1)$. Suppose $\{\hat{y}_i\}_{i\in[n]}$ is an optimal solution to (REL-W-3) for $B' = \lfloor \frac{B}{p^m} \rfloor$, $\boldsymbol{u}_i \boldsymbol{u}_i^\top = p^{m-k_i} q^{k_i} \boldsymbol{v}_i \boldsymbol{v}_i^\top$.
for $j = 1, \ldots, \lfloor \frac{B}{p^m} \rfloor$ **do**
    Sample s from $[n]$ with probability $\mathbb{P}[s = i] = \frac{\hat{y}_i}{\lfloor \frac{B}{p^m} \rfloor}$.
    Let $\tilde{S} = \tilde{S} \cup \{ \underbrace{s, s, \ldots, s}_{p^{m-k_s} q^{k_s} \text{ copies}} \}$.
end for.
while $c(\tilde{S}) \leq B - c_j$ for some $j \in [n]$ **do** ▷ An optional step to improve the objective.
    Let $j^* \in \operatorname{argmax}_{j\in[n], c(\tilde{S}) < B - c_j} c_j^{-1} [\det(\sum_{i\in\tilde{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top + \boldsymbol{v}_j \boldsymbol{v}_j^\top) - \det(\sum_{i\in\tilde{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top)]$.
    Add j^* to multi-set $\tilde{S}$.
end while.
Return $\tilde{S}$.

The same reference also proves the polynomial-time complexity of the algorithm, and describes a deterministic implementation with the same guarantee.

The approximation guarantee for Algorithm 3 is obtained from Theorem 5 of Singh & Xie (2020). The only missing piece is to relate the optimal value $w_3(\left\lfloor \frac{B}{p^m} \right\rfloor)$ of (REL-W-3) to the optimal value $w_2(\frac{B}{p^m})$ of (REL-W-2). This is done in the following technical lemmas.

Lemma 5. For any $B', B'' > 0$, we have

$$w_2\left(B'\right) = w_2\left(B''\right) + \log\left(\frac{B'}{B''}\right), \quad w_3\left(B'\right) = w_3\left(B''\right) + \log\left(\frac{B'}{B''}\right).$$

Proof. The two equalities hold because, for any feasible solution $\boldsymbol{y}$ to (REL-W-2) or (REL-W-3) with budget B', we can construct a feasible solution to (REL-W-2) or (REL-W-3) with budget B'' by letting $\hat{y}_i = \frac{B''}{B'} y_i$ for each $i \in [n]$ and vice versa. □

Lemma 6. The following inequalities must hold:

$$w_2\left(\frac{B}{p^m}\right) \geq w_3\left(\left\lfloor \frac{B}{p^m} \right\rfloor\right) \geq w_2\left(\frac{B}{p^m}\right) - \log\left(\frac{p}{q} \frac{\frac{B}{p^m}}{\left\lfloor \frac{B}{p^m} \right\rfloor}\right).$$

Proof. The first inequality holds because $\frac{q}{p} < \tilde{c}_i \leq 1$ for each $i \in [n]$ and $B'' \leq B'$. To prove the second inequality, we replace each $\tilde{c}_i$ in (REL-W-2) by its lower bound $\frac{q}{p}$, and let $B' = \frac{q}{p} \left\lfloor \frac{B}{p^m} \right\rfloor$,

yielding

$$w_3\left(\left\lfloor\frac{B}{p^m}\right\rfloor\right) \geq w_2\left(\frac{q}{p}\left\lfloor\frac{B}{p^m}\right\rfloor\right).$$

By Lemma 5, we have

$$w_2\left(\frac{q}{p}\left\lfloor\frac{B}{p^m}\right\rfloor\right) = w_2\left(\frac{B}{p^m}\right) + \log\left(\frac{q}{p}\frac{\left\lfloor\frac{B}{p^m}\right\rfloor}{\frac{B}{p^m}}\right),$$

which completes the proof □

The main result then follows by combining Lemma 5, Lemma 6, (10), and Theorem 5 of Singh & Xie (2020). The proof is straightforward and thus omitted.

Theorem 5. Algorithm 3 yields a $\frac{1}{e}\frac{q}{p}\frac{B-p^m}{B}$-approximation, i.e.,

$$\left(\mathbb{E}\left[\det\left(\sum_{i\in\tilde{S}}\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)\right]\right)^{\frac{1}{d}} \geq \frac{1}{e}\frac{q}{p}\frac{B-p^m}{B}\det\left(\sum_{i\in[n]}\hat{x}_i\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)^{\frac{1}{d}}.$$

In particular, when $B \geq q^m\frac{c_{\max}}{\epsilon}$ for some $\epsilon \in (0, \frac{q}{p})$, we have

$$\left(\mathbb{E}\left[\det\left(\sum_{i\in\tilde{S}}\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)\right]\right)^{\frac{1}{d}} \geq \frac{1}{e}\left(\frac{q}{p}-\epsilon\right)\det\left(\sum_{i\in[n]}\hat{x}_i\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)^{\frac{1}{d}}.$$

Furthermore, given $\epsilon \in (0, \frac{q}{p})$ and $B \geq p^{\lceil\log_{\frac{p}{q}}(c_{\max})\rceil}\frac{d}{\epsilon}$, Algorithm 3 yields a $(1-\frac{\epsilon}{d})(\frac{q}{p}-\epsilon)$-approximation, i.e.,

$$\left(\mathbb{E}\left[\det\left(\sum_{i\in\tilde{S}}\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)\right]\right)^{\frac{1}{d}} \geq \left(1-\frac{\epsilon}{d}\right)\left(\frac{q}{p}-\epsilon\right)\det\left(\sum_{i\in[n]}\hat{x}_i\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)^{\frac{1}{d}}.$$

The time complexity of SA-W depends mainly on the cost of computing the continuous relaxation. According to Ben-Tal & Nemirovski (2001), this cost is $O(n^5)$. After solving the continuous relaxation, SA-W incurs a cost of $O(n)$ per iteration. As the number of iterations is $\lfloor\frac{B}{p^m}\rfloor$, the expected running time is $O(n^5 + n\lfloor\frac{B}{p^m}\rfloor)$, as reported in Table 1.

Algorithm 4 Sampling algorithm for budget-constrained D-optimal design without repetition.

Initialize $\tilde{S} = \emptyset$ and $\epsilon \in (0, 1)$. Suppose $\hat{\boldsymbol{x}}$ is an optimal solution to the continuous relaxation of (MIP-WO).
do
  $\tilde{S} = \emptyset$
  for $j \in [n]$ **do**
    Sample a $(0, 1)$ uniform random variable U.
    if $U \leq \frac{\hat{x}_j}{1+\epsilon}$ **then**
      Add j to the set $\tilde{S}$.
    end if.
  end for.
while $c(\tilde{S}) > B$.
while $c(\tilde{S}) \leq B - c_j$ for some $j \notin \tilde{S}$ **do** ▷ Optional step to improve the objective.
  Let $j^* \in \underset{j: j \in [n] \setminus \tilde{S}, c(\tilde{S}) < B - c_j}{\operatorname{argmax}} c_j^{-1}[\det(\sum_{i \in \tilde{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top + \boldsymbol{v}_j \boldsymbol{v}_j^\top) - \det(\sum_{i \in \tilde{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top)]$.
  Add j^* to set $\tilde{S}$.
end while.

6 Sampling without repetition (SA-WO)

Algorithm 4 describes a sampling method without repetition. We first compute an optimal solution $\hat{\boldsymbol{x}}$ to the continuous relaxation of (MIP-WO) using the Frank-Wolfe algorithm or another convex optimization technique. Next, given $\epsilon > 0$ (the precise value will be optimized for the purpose of proving the approximation guarantee), we select experiment j with probability $\frac{\hat{x}_j}{1+\epsilon}$ independently. Let $\tilde{S}$ be the index set of experiments selected in this manner. If $c(\tilde{S}) > B$, we discard $\tilde{S}$ and start over. If $c(\tilde{S}) < B - c_j$ for some $j \notin \tilde{S}$, we add more experiments to use more of the budget.

Section 6.1 presents the approximation guarantee for Algorithm 4. Section 6.2 gives a deterministic version of the algorithm, which is the most novel development in this section, as the derandomization technique of Singh & Xie (2020) no longer works in the budget-constrained setting. Similarly to SA-W, the time complexity of SA-WO is dominated by the cost of solving the continuous relaxation, which is $O(n^5)$ as reported in Table 1.

6.1 Approximation guarantee

We first state the guarantee. It is worth noting that, although we assume $\min_{i \in [n]} c_i = 1$, the results in Theorem 6 do not depend on the ratio $\max_{i \in [n]} c_i / \min_{i \in [n]} c_i$, which appears in the definition of m used in our other algorithms.

Theorem 6. Given an $\epsilon \in (0, 1)$, suppose that $B \geq \frac{4 c_{\max} d}{\epsilon} + \frac{12 c_{\max}}{\epsilon^2} \log(\frac{1}{\epsilon})$ and $\tilde{S}$ denotes the output

of Algorithm 4. Then, $\tilde{S}$ satisfies

$$\left(\mathbb{E}\left[\det\left(\sum_{i\in\tilde{S}}\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)\right]\right)^{\frac{1}{d}} \geq (1-\epsilon)\det\left(\sum_{i\in[n]}\hat{x}_i\boldsymbol{v}_i\boldsymbol{v}_i^\top\right)^{\frac{1}{d}}.$$

The proof of 6 proceeds very similarly to Singh & Xie (2020), where an analogous guarantee is derived by combining the Cauchy-Binet formula (Lemma 3 in Singh & Xie 2020) with a property of the distribution of the random set $\tilde{S}$ returned by the sampling method. The distribution is required to be d-wise α-positively correlated, meaning that

$$\mathbb{P}\left[T\subseteq\tilde{S}|c(\tilde{S})\leq B\right]\geq\alpha^d\prod_{i\in T}\hat{x}_i$$

for some $\alpha>0$. Therefore, our analysis here essentially reduces to proving that this property holds in the context of heterogeneous costs. This gap is filled by the following technical lemma.

Lemma 7. Let $\tilde{S}$ be a random set obtained from one iteration of the while loop described in lines 3-9) of Algorithm 4. Given $T\subseteq[n]$ with $|T|=d\leq n$ and $\epsilon\in(0,1)$, we have

$$\frac{1}{\prod_{i\in T}\hat{x}_i}\mathbb{P}\left[T\subseteq\tilde{S}|c(\tilde{S})\leq B\right]\geq(1+\epsilon)^{-d}\left(1-e^{-\frac{(\epsilon B-(1+\epsilon)c(T))^2}{c_{\max}B(2+\epsilon)(1+\epsilon)}}\right):=\alpha^d.$$

The proof of Lemma 7 uses a weighted Chernoff bound implied by Corollary 1 in Panconesi & Srinivasan (1997). The detailed proof can be found in Appendix B.3.

Lemma 8 (Weighted Chernoff bound). Let $X_1,...,X_n$ be independent 0/1 random variables, each with a weight $0\leq w_1,...,w_n\leq 1$, and let X be the weighted sum $X=w_1X_1+\ldots+w_nX_n$. For $\mu=\mathbb{E}[X]$ and every $\bar{\epsilon}>0$, we have

$$\mathbb{P}\left[X\geq(1+\bar{\epsilon})\,\mu\right]\leq e^{-\frac{\bar{\epsilon}^2}{2+\bar{\epsilon}}\mu}.$$

Finally, we comment on the computational complexity of Algorithm 4. The sampling procedure at each iteration of the while loop has complexity $O(n)$. Under the same conditions as Theorem 6, we show that Algorithm 4 terminates after a constant number of iterations in expectation. The proof is again based on the weighted Chernoff bound of Lemma 8; the details can be found in

Appendix B.3.

Proposition 6. Given $\epsilon \in (0,1)$, suppose that $B > \frac{4c_{\max}d}{\epsilon} + \frac{12c_{\max}}{\epsilon^2}\log(\frac{1}{\epsilon})$. Then the expected number of iterations (lines 3-9) in Algorithm 4 is $O(1)$.

6.2 Deterministic implementation

Our results for Algorithm 4 hold in expectation, and one may have to sample multiple times in order to obtain a feasible solution. We propose a derandomized (deterministic) version of Algorithm 4 that returns a feasible solution with the same guarantee. While Singh & Xie (2020) also presents deterministic implementations of sampling algorithms for the cardinality-constrained case, our developments here are significantly different.

To derandomize Algorithm 4, we use conditional expectation as in Spencer (1994). Let $S \subseteq [n]$ satisfy $c(S) \leq B$, and let $U \subseteq [n] \setminus S$. One may view S as an index set of experiments that must be selected, while U is an index set of experiments that must not be selected. We then define

$$H(S,U) := \mathbb{E}\left[\det\left(\sum_{i\in\tilde{S}} \boldsymbol{v}_i\boldsymbol{v}_i^\top\right)\middle| S \subseteq \tilde{S} \subseteq [n]\setminus U, c(\tilde{S}) \leq B\right]$$

to be the conditional expectation of the objective value achieved by Algorithm 4, given that the experiments in S were sampled while the experiments in U were not. This definition generalizes what is used in Singh & Xie (2020) for the cardinality-constrained case, where the conditional expectation was a function of S only.

Observe that $H(\emptyset,\emptyset)$ is simply the expected output value of Algorithm 4. Suppose that S and U are chosen in such a way that there exists a $j \in [n]\setminus(S\cup U)$ satisfying $c(S) + c_j \leq B$. Then, we must have

$$\begin{aligned} H(S,U) = {} & H(S\cup j, U)\Pr\left[j \in \tilde{S}\middle| S\subseteq\tilde{S}, c(\tilde{S}) \leq B\right] \\ & + H(S, U\cup j)\Pr\left[j \notin \tilde{S}\middle| S\subseteq\tilde{S}, c(\tilde{S}) \leq B\right]. \end{aligned}$$

Thus, $\max\{H(S\cup j, U), H(S, U\cup j)\} \geq H(S,U)$, due to the fact that

$$\Pr\left[j \in \tilde{S}\middle| S\subseteq\tilde{S}, c(\tilde{S}) \leq B\right] + \Pr\left[j \notin \tilde{S}\middle| S\subseteq\tilde{S}, c(\tilde{S}) \leq B\right] = 1.$$

Algorithm 5 Derandomization of Algorithm 4

Suppose $\hat{\boldsymbol{x}}$ is an optimal solution to the continuous relaxation of (MIP-WO).
Initialize $S = \emptyset$, $U = \emptyset$ and $j = 1$.
do
    if $H(S \cup j, U) \geq H(S, U \cup j)$ **then**
        $S = S \cup j$
    else
        $U = U \cup j$
    end if
while there exist a $j \in [n] \setminus (S \cup U)$ such that $c(S) + c_j \leq B$.
Return S.

On the other hand, if S and U are chosen in such a way that there is no $j \in [n] \setminus (S \cup U)$ satisfying $c(S) + c_j \leq B$, we have

$$H(S, U) = \det\left(\sum_{i \in S} \boldsymbol{v}_i \boldsymbol{v}_i^\top\right),$$

where the equality holds because the only $\tilde{S}$ satisfying $S \subseteq \tilde{S} \subseteq [n] \setminus U$ and $c(\tilde{S}) \leq B$ is $\tilde{S} = S$.

These observations motivate the structure of the deterministic algorithm, stated formally as Algorithm 5. In each iteration, we evaluate some $j \in [n] \setminus (S \cup U)$ for which $c(S) + c_j \leq B$. If $H(S \cup j, U) \geq H(S, U \cup j)$, we add j to the set S; otherwise, we add j to the set U. Because we start with $S = U = \emptyset$, this procedure will yield S whose objective value is greater than or equal to $H(\emptyset, \emptyset)$, which is the expected value of the solution returned by Algorithm 4. Thus, the solution found by Algorithm 5 in a deterministic manner is guaranteed to be better than the expected value of the output of the sampling algorithm. It follows that the approximation guarantee of Theorem 6 always holds for the output of Algorithm 5.

The only remaining issue is the efficient computation of $H(S, U)$. Since U consists of elements that are removed from $[n]$, the cost of computing $H(S, U)$ is bounded above by the cost of computing $H(S, \emptyset)$. It can be shown, using arguments similar to those used in Singh & Xie (2020), that this evaluation can be reduced to efficient computations of two formulas. However, instead of computing the coefficient of a single term in a polynomial, as is done in Singh & Xie (2020), we have to compute the sum of coefficients of multiple terms in a "generalized polynomial" that has (possibly) non-integer exponents. Specifically, the two formulas to be computed are (i) the sum of coefficients with order less than or equal to $B - c(S)$ in the polynomial $P(y) = \prod_{i \in [n] \setminus S} (1 + \frac{\hat{x}_i}{1+\epsilon-\hat{x}_i} y^{c_i})$, which can be efficiently computed using a fast Fourier transform, and (ii) the sum of coefficients of terms of the

form $t_1^d t_2^{\tilde{B}-c(S)} t_3^{c(S)}$, for any $\tilde{B}$ satisfying $c(S) \leq \tilde{B} \leq B$, in the expansion of

$$F(t_1, t_2, t_3) = \det\left(\boldsymbol{I}_n + t_1 \operatorname{diag}(\boldsymbol{y})^{\frac{1}{2}} \boldsymbol{A}^\top \boldsymbol{A} \operatorname{diag}(\boldsymbol{y})^{\frac{1}{2}} + \operatorname{diag}(\boldsymbol{y})\right),$$

where $t_1, t_2, t_3 \in \mathbb{R}, \boldsymbol{y} \in \mathbb{R}^n$ are indeterminate and

$$y_i = \begin{cases} t_3^{c_i} & \text{if } i \in S \\ \frac{\hat{x}_i}{1+\epsilon-\hat{x}_i} t_2^{c_i} & \text{otherwise.} \end{cases}$$

This computation can be efficiently performed using the Faddeev-LeVerrier algorithm (Hou 1998). Having observed that these formulas can be computed efficiently, one can then extend the proofs in Singh & Xie (2020) to conclude that each evaluation of $H(\cdot, \cdot)$ can be performed efficiently at cost of $O(n^4)$. Algorithm 5 requires at most $O(n)$ evaluations of the function $H(\cdot, \cdot)$. Therefore, the total complexity of Algorithm 5 is $O(n^5)$.

7 Numerical study

We present empirical insights into the performance of our algorithms. Section 7.1 describes the synthetic datasets that we used, and Section 7.2 presents the results, discussing both solution quality and computational cost.

7.1 Data and experimental setup

We generated two sets of experiments, a "small" one with $n = 300$ and $d = 14$, and a "large" one with $n = 1000$ and $d = 49$. The vectors $\boldsymbol{v}_i$ in each dataset contain d_1 continuous elements (with $d_1 = 9$ in the small dataset and $d_1 = 30$ in the large one) and $d - d_1$ binary-valued elements based on a single categorical variable. The continuous elements are generated by sampling from a uniform distribution on the unit sphere in $\mathbb{R}^{d_1}$. To generate the dummy variables, we first sample $d - d_1 + 1$ random numbers from an exponential distribution with parameter 1, and then normalize them to create a probability mass function (essentially this is a uniform sample on the unit simplex). From this PMF, we generate a category that determines the values of the dummy variables.

The costs c_i were sampled from a uniform distribution on the interval $[1, b]$ where two values $b \in \{2, 16\}$ were considered for both datasets. Thus, we have a total of eight experimental settings:

two data sizes, two choices of b, and with/without repetition. For each of these settings, we generated 5 random instances based on the above specifications. Within each instance, we ran seven different budgets (values of B), starting from a multiple of 50 that exceeded the thresholds required by our approximation guarantees.

We report the average performance of each algorithm, measured using the log-determinant $\frac{1}{d}\log\det(\cdot)$, across these instances. Since the output of SA-W and SA-WO is random, we ran these algorithms 20 times on each instance and report the best-performing solution (which is then averaged across the instances). Since it is difficult to solve (MIP-W) or (MIP-WO) to optimality, we compare the performance of each algorithm to the upper bound provided by solving the continuous relaxation of each problem. We also report the total running time of each algorithm.

Although, for three of our four algorithms, the approximation guarantees depend on parameters p, q, we found that $p = 2, q = 1$ produced consistently good results. For simplicity, we used these values in all numerical instances. We also used $\epsilon = 0$ in the implementation of SA-WO.

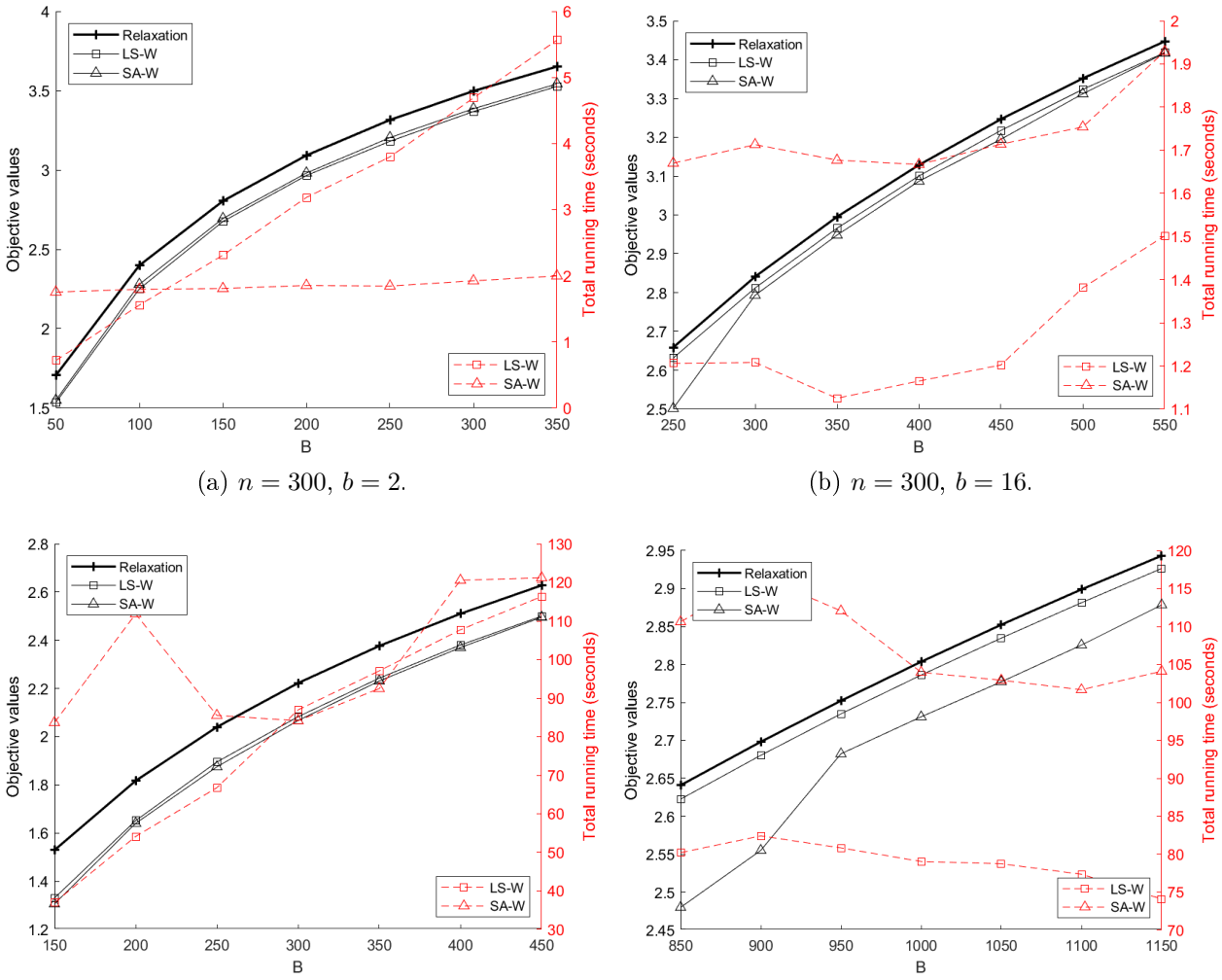
When repetition is not allowed, we also implement the RE method of Lau & Zhou (2022). Since RE is also randomized, we ran it 20 times on each instance similarly to SA-W and SA-WO. RE is not guaranteed to produce a feasible solution; if it was not able to find one in 20 attempts, we report the least infeasible one (i.e., the solution whose cost is closest to the budget).

7.2 Numerical results

Figures 1-2 report results for the eight experimental settings. The four settings with repetition are shown in Figure 1, while the settings without repetition are presented in Figure 2. In all cases, the optimality gaps for all methods shrink as B becomes larger, consistent with the asymptotic behavior of the approximation guarantees.

The RE method is not shown in Figure 1 as it does not handle repetition. LS-W tends to outperform SA-W, in both objective value and computational cost, except for the case $n = 300$, $b = 2$. Even in that case, the good performance of SA-W is due to the optional greedy step in lines 6-9 of Algorithm 3. If this step is omitted, LS-W will perform better. In the other cases, LS-W performs better even when SA-W is given the benefit of the optional step.

For D-optimal design with repetition, LS-W has a higher objective value than SA-W except for when $n = 300, b = 2$ and the budget B is small. Both their optimality gaps based on the optimal



(a) $n = 300$, $b = 2$.

(b) $n = 300$, $b = 16$.

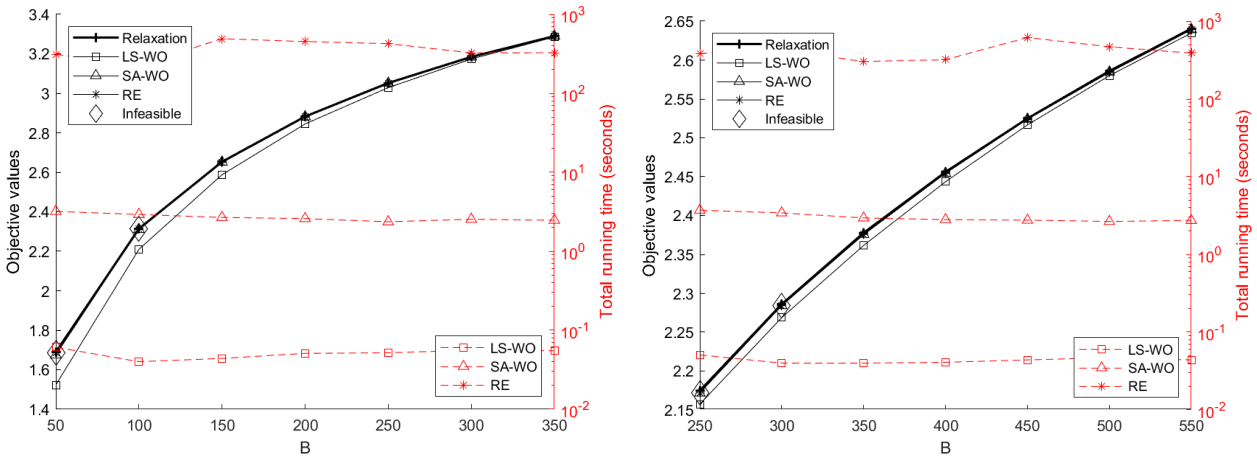
(c) $n = 1000$, $b = 2$.

(d) $n = 1000$, $b = 16$.

Figure 1: Objective values and total running time for the D-optimal design with repetition.

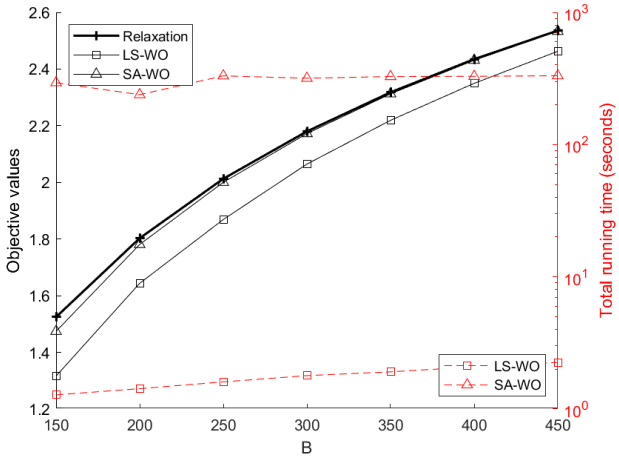
continuous relaxation value are small. Although LS-W is worse than SA-W when $n = 300, b = 2$, from our experiments not shown in the figures, if we compare the solutions before the "greedy addition" at the end of the algorithms, LS-W generally has a better objective value than that of SA-W. We also note that the reported performance of SA-W is the best obtained over 20 rounds of sampling.

With regard to the running time, LS-W becomes more expensive for higher B, because there are now more possible interchanges to search over, and, since repetition is allowed, the number of possible interchanges does not decrease as the algorithm progresses. The cost of SA-W does not depend on B much because the time complexity of this method is dominated by the cost of solving the continuous relaxation.

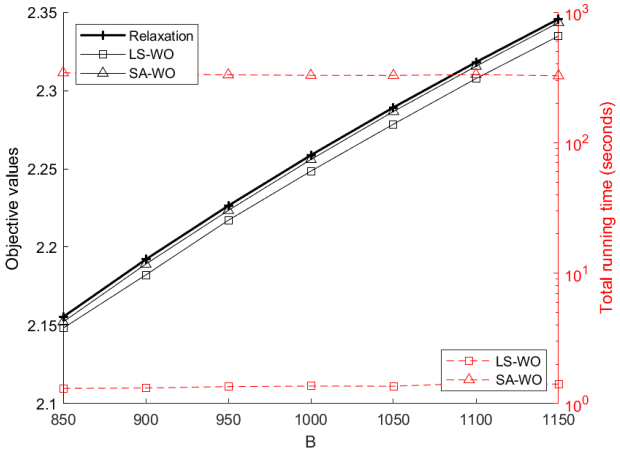


(a) $n = 300$, $b = 2$.

(b) $n = 300$, $b = 16$.



(c) $n = 1000$, $b = 2$.



(d) $n = 1000$, $b = 16$.

Figure 2: Objective values and total running time for the D-optimal design without repetition.

We now turn to Figure 2. First, it is necessary to mention that, for the instances with $n = 1000$, the RE benchmark did not terminate within 5 hours (not even with an infeasible solution). This may be due to the fact that the approximation guarantee of RE requires a much larger B than our methods; another reason could be that RE has to solve an algebraic decomposition problem at every iteration, incurring significant computational cost. Thus, we were not able to include RE in the comparison for $n = 1000$. When $n = 300$, we found that RE performed similarly to SA-WO but with much longer running time. Furthermore, for the two lowest B values, RE was not able to find a feasible solution in 20 attempts.

As for LS-WO, we found that it underperformed SA-WO in problems without repetition, but also ran significantly faster. For the instances with $n = 1000$, SA-WO can take 200 times longer to

terminate than LS-WO. The reason is that the potential function of LS-WO can be computed much more efficiently than the one used by LS-W. This is an important benefit because the computational cost often increases across the board when repetition is not allowed (since we then have to examine combinations of experiments rather than simply making copies of the same one).

These comparisons have revealed some of the strengths and weaknesses of each approach. Local search tends to outperform sampling when repetition is allowed, and if it is not allowed, the situation is reversed. However, local search is consistently the most computationally efficient approach, and, in particular, scales much better with large n.

8 Conclusion

We have presented a suite of polynomial-time algorithms for the budget-constrained D-optimal design problem, based on local search and sampling. Two variants of each approach are developed in order to handle situations where repetition is and is not allowed. An approximation guarantee is proved for each algorithm, and deterministic implementations are provided for the sampling algorithms. Our deterministic approximation ratios for the budget-constrained D-optimal design problem fill a significant gap in the existing literature, which heretofore has mainly focused on the cardinality-constrained case; the only previously existing method for the budget-constrained problem relies on probabilistic guarantees and may not produce a feasible solution. Numerical experiments show the practical viability of our algorithms.

A novel aspect of our work is the idea of grouping the experiments into packs of similar cost, which has not previously been considered by the literature on D-optimal design. Our rounding scheme and interchange rule could potentially have uses in other classes of budget-constrained combinatorial optimization problems.

Acknowledgments

The first and third authors gratefully acknowledge the support of NSF grant CMMI-2112828. The second author gratefully acknowledges the support of NSF grant CCF-2246417.

References

Ahipaşaoğlu, S. D. (2021), 'A branch-and-bound algorithm for the exact optimal experimental design problem', *Statistics and Computing* **31**(65), 65:1–65:11.

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

Anstreicher, K. M. (2020), 'Efficient solution of maximum-entropy sampling problems', *Operations Research* **68**(6), 1826–1835.

Atkinson, A., Donev, A. & Tobias, R. (2007), *Optimum experimental designs, with SAS*, Vol. 34, Oxford University Press.

Ben-Tal, A. & Nemirovski, A. (2001), *Lectures on modern convex optimization*, Society for Industrial and Applied Mathematics.

Bouhtou, M., Gaubert, S. & Sagnol, G. (2010), 'Submodularity and randomized rounding techniques for optimal experimental design', *Electronic Notes in Discrete Mathematics* **36**, 679–686. ISCO 2010 - International Symposium on Combinatorial Optimization.

Brown, A., Laddha, A., Pittu, M. & Singh, M. (2022), 'Efficient determinant maximization for all matroids', *ArXiv*.

Coey, C., Kapelevich, L. & Vielma, J. P. (2023), 'Conic optimization with spectral functions on Euclidean Jordan algebras', *Mathematics of Operations Research* **48**(4), 1906–1933.

de Aguiar, P., Bourguignon, B., Khots, M., Massart, D. & Phan-Than-Luu, R. (1995), 'D-optimal designs', *Chemometrics and Intelligent Laboratory Systems* **30**(2), 199–210.

Horel, T., Ioannidis, S. & Muthukrishnan, S. (2014), Budget feasible mechanisms for experimental design, *in* A. Pardo & A. Viola, eds, 'LATIN 2014: Theoretical Informatics', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 719–730.

Hou, S.-H. (1998), 'Classroom note: A simple proof of the Leverrier–Faddeev characteristic polynomial algorithm', *SIAM Review* **40**(3), 706–709.

Huang, C.-C. & Kakimura, N. (2021), 'Improved streaming algorithms for maximizing monotone submodular functions under a knapsack constraint', *Algorithmica* **83**, 879–902.

Ko, C.-W., Lee, J. & Queyranne, M. (1995), 'An exact algorithm for maximum entropy sampling', *Operations Research* **43**(4), 684–691.

Lau, L. C. & Zhou, H. (2022), 'A local search framework for experimental design', *SIAM Journal on Computing* **51**(4), 900–951.

Li, Y. & Xie, W. (2023), 'Best principal submatrix selection for the maximum entropy sampling problem: scalable algorithms and performance guarantees', *Operations Research (to appear)*.

Li, Y., Fampa, M., Lee, J., Qiu, F., Xie, W. & Yao, R. (2024), 'D-optimal data fusion: exact and approximation algorithms', *INFORMS Journal on Computing* **36**(1), 97–120.

Madan, V., Nikolov, A., Singh, M. & Tantipongpipat, U. (2020), Maximizing determinants under matroid constraints, *in* '2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)', pp. 565–576.

Madan, V., Singh, M., Tantipongpipat, U. & Xie, W. (2019), Combinatorial algorithms for optimal design, *in* A. Beygelzimer & D. Hsu, eds, 'Proceedings of the Thirty-Second Conference on Learning Theory', Vol. 99 of *Proceedings of Machine Learning Research*, PMLR, pp. 2210–2258.

Mahabadi, S. & Vuong, T. D. (2022), 'Composable coresets for constrained determinant maximization and beyond', *ArXiv*.

Nikolov, A. (2015), Randomized rounding for the largest simplex problem, *in* 'Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing', STOC '15, Association for Computing Machinery, New York, NY, USA, p. 861–870.

Nikolov, A., Singh, M. & Tantipongpipat, U. T. (2022), 'Proportional volume sampling and approximation algorithms for A-optimal design', *Mathematics of Operations Research* **47**(2), 847–877.

Panconesi, A. & Srinivasan, A. (1997), 'Randomized distributed edge coloring via an extension of the Chernoff–Hoeffding bounds', *SIAM Journal on Computing* **26**(2), 350–368.

Pokhilko, V., Zhang, Q., Kang, L. & Mays, D. P. (2019), 'D-optimal design for network A/B testing', *Journal of Statistical Theory and Practice* **13**(4), 65:1–65:23.

Ponte, G., Fampa, M. & Lee, J. (2023), 'Branch-and-bound for D-optimality with fast local search and variable-bound tightening', *ArXiv*.

Pukelsheim, F. (2006), *Optimal design of experiments*, Society for Industrial and Applied Mathematics.

Ravi, S. N., Ithapu, V., Johnson, S. & Singh, V. (2016), Experimental design on a budget for sparse linear models and applications, *in* 'International conference on machine learning', PMLR, pp. 583–592.

Sagnol, G. & Harman, R. (2015), 'Computing exact D-optimal designs by mixed integer second-order cone programming', *The Annals of Statistics* **43**(5), 2198–2224.

Sansone, M., Fusco, R. & Petrillo, A. (2019), 'D-optimal design of b-values for precise intra-voxel incoherent motion imaging', *Biomedical Physics & Engineering Express* **5**(3), 035025:1–035025:14.

Singh, M. & Xie, W. (2020), 'Approximation algorithms for D-optimal design', *Mathematics of Operations Research* **45**(4), 1512–1534.

Spencer, J. (1994), *Ten lectures on the probabilistic method*, 2nd edition edn, Society for Industrial and Applied Mathematics.

Sviridenko, M. (2004), 'A note on maximizing a submodular set function subject to a knapsack constraint', *Operations Research Letters* **32**(1), 41–43.

Tie, L., Cai, K.-Y. & Lin, Y. (2011), 'Rearrangement inequalities for Hermitian matrices', *Linear Algebra and its Applications* **434**(2), 443–456.

Vandenberghe, L., Boyd, S. & Wu, S.-P. (1998), 'Determinant maximization with linear matrix inequality constraints', *SIAM Journal on Matrix Analysis and Applications* **19**(2), 499–533.

Wang, J., Xie, W., Ryzhov, I. O., Markovic, N. & Ou, G. (2023), 'D-optimal orienteering for post-earthquake reconnaissance planning', *Working paper, University of Maryland.*

Yaroslavtsev, G., Zhou, S. & Avdiukhin, D. (2020), "Bring your own greedy"+max: near-optimal 1/2-approximations for submodular knapsack, *in* S. Chiappa & R. Calandra, eds, 'Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics', Vol. 108 of *Proceedings of Machine Learning Research*, PMLR, pp. 3263–3274.

Zhu, L., Dasgupta, T. & Huang, Q. (2014), 'A D-optimal design for estimation of parameters of an exponential-linear growth curve of nanostructures', *Technometrics* **56**(4), 432–442.

A Subtle search

In this section, we present the subtle search procedure, which can improve the numerical performance of Algorithms 1 and 2. The theoretical guarantee will remain unchanged. We discuss the differences between subtle and ordinary search and present subtle search formally in Algorithms 6 and 7. In the following, we focus on the case where repetition is allowed; the discussion for the other case is similar and we omit it for brevity.

Interchange rules. Recall that, in LS-W, each interchange made by ordinary search replaces $p^{m-r}q^r$ experiments with cost a^r by $p^{m-\ell}q^l$ number of experiments with cost a^ℓ. Subtle search involves fewer experiments in each interchange. Specifically, for any two different exponents $r, \ell \in M$, we consider two cases: (i) if $r > \ell$, we can replace $q^{r-\ell}$ experiments with cost a^r by $p^{r-\ell}$ copies of an experiment of cost a^ℓ; and (ii) if $r \leq \ell$, we can replace $p^{\ell-r}$ experiments with cost a^r by $q^{\ell-r}$ copies of an experiment with cost a^ℓ. It is critical to apply ordinary search first to obtain the theoretical guarantee, since subtle search alone does not have any theoretical properties.

Properties. The result of Proposition 1 holds for subtle search with a nearly identical proof. The time complexity for each interchange of subtle search is bounded above by the complexity derived in Proposition 2. Both ordinary search and subtle search require the same amount of computation, but fewer experiments are involved at each interchange of subtle search.

Algorithm 6 *SubtleSearchW*$(\tilde{S}, \boldsymbol{X}_{\tilde{S}}, \boldsymbol{\Lambda})$

1: **while** $c(\tilde{S}) \le B - c_j$ for some $j \in [n]$ **do**
2: Let $j^* \in \operatorname{argmax}_{j\in[n], c(\tilde{S})<B-c_j} c_j^{-1}[\det(\sum_{i\in\tilde{S}} \boldsymbol{v}_i\boldsymbol{v}_i^\top + \boldsymbol{v}_j\boldsymbol{v}_j^\top) - \det(\sum_{i\in\tilde{S}} \boldsymbol{v}_i\boldsymbol{v}_i^\top)]$.
3: Add j^* to set $\tilde{S}$.
4: **end while**
5: **for** $r, \ell \in M$ **do**
6: $\tilde{S}_r = \{i \in \tilde{S} : c_i = a^r\}$
7: **if** $\tilde{S}_r \neq \emptyset$ **then**
8: **if** $r > \ell$ and $|\{i \in \tilde{S} : c_i = a^r\}| \ge q^{r-\ell}$ **then**
9: $J_{r,\ell}, I_{r,\ell} = \operatorname*{argmax}_{J,I}\{f(J,I) : |I| = q^{r-\ell}, I \subseteq \tilde{S}_r, c_j = a^\ell, J = \{\underbrace{j, \cdots, j}_{p^{r-\ell} \text{ copies}}\}\}$.
10: **end if**
11: **if** $r \le \ell$ and $|\{i \in \tilde{S} : c_i = a^r\}| \ge p^{\ell-r}$ **then**
12: $J_{r,\ell}, I_{r,\ell} = \operatorname*{argmax}_{J,I}\{f(J,I) : |I| = p^{\ell-r}, I \subseteq \tilde{S}_r, c_j = a^\ell, J = \{\underbrace{j, \cdots, j}_{q^{\ell-r} \text{ copies}}\}\}$.
13: **end if**
14: **end if**
15: **end for**
16: $r_0, \ell_0 = \operatorname{argmax}_{r,\ell} f(J_{r,\ell}, I_{r,\ell})$.
17: **if** $f(J_{r_0,\ell_0}, I_{r_0,\ell_0}) > 0$ **then**
18: Add the multi-set J_{r_0,ℓ_0} to $\tilde{S}$ and delete the multi-set I_{r_0,ℓ_0} from $\tilde{S}$.
19: $\boldsymbol{X}_{\tilde{S}} = \boldsymbol{X}_{\tilde{S}} + \sum_{j\in J_{r_0,\ell_0}} \boldsymbol{v}_j\boldsymbol{v}_j^\top - \sum_{i\in I_{r_0,\ell_0}} \boldsymbol{v}_i\boldsymbol{v}_i^\top$.
20: **for** $j \in J_{r_0,\ell_0}$ **do**
21: $\boldsymbol{\Lambda} = \boldsymbol{\Lambda} - \frac{\boldsymbol{\Lambda}\boldsymbol{v}_j\boldsymbol{v}_j^\top\boldsymbol{\Lambda}}{1+\boldsymbol{v}_j^\top\boldsymbol{\Lambda}\boldsymbol{v}_j}$
22: **end for**
23: **for** $i \in I_{r_0,\ell_0}$ **do**
24: $\boldsymbol{\Lambda} = \boldsymbol{\Lambda} + \frac{\boldsymbol{\Lambda}\boldsymbol{v}_i\boldsymbol{v}_i^\top\boldsymbol{\Lambda}}{1-\boldsymbol{v}_i^\top\boldsymbol{\Lambda}\boldsymbol{v}_i}$
25: **end for**
26: Update $\tau_{ij} = \boldsymbol{v}_i^\top\boldsymbol{\Lambda}\boldsymbol{v}_j$ for all $i \in \tilde{S}$, $j \in [n]$, and $\tau_j = \boldsymbol{v}_j^\top\boldsymbol{\Lambda}\boldsymbol{v}_j$ for all $j \in [n]$.
27: Return to line 5.
28: **end if**
29: **return** $\tilde{S}, \boldsymbol{X}_{\tilde{S}}$.

Algorithm 7 *SubtleSearchWO*$(\tilde{S}, \boldsymbol{X}_{\tilde{S}}, \boldsymbol{\Lambda})$

1: **while** $c(\tilde{S}) \leq B - c_j$ for some $j \notin \tilde{S}$ **do**
2: Let $j^* \in \operatorname{argmax}_{j \notin \tilde{S}, c(\tilde{S}) < B - c_j} [\frac{\det(\sum_{i \in \tilde{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top + \boldsymbol{v}_j \boldsymbol{v}_j^\top) - \det(\sum_{i \in \tilde{S}} \boldsymbol{v}_i \boldsymbol{v}_i^\top)}{c_j}]$.
3: Add j^* to set $\tilde{S}$.
4: **end while**
5: **for** $r, \ell \in M$ **do**
6: $\tilde{S}_r = \{i \in \tilde{S} : c_i = a^r\}$
7: **if** $r > \ell$ and $|\{i \in \tilde{S} : c_i = a^r\}| \geq q^{r-\ell}$ **then**
8: $J_{r,\ell}, I_{r,\ell} = \operatorname*{argmax}_{J,I} \{f(J, I) : |I| = q^{r-\ell}, I \subseteq \tilde{S}_r, |J| = p^{r-\ell}, J \subseteq \tilde{S}_\ell^C\}$.
9: **end if**
10: **if** $r \leq \ell$ and $|\{i \in \tilde{S} : c_i = a^r\}| \geq p^{\ell-r}$ **then**
11: $J_{r,\ell}, I_{r,\ell} = \operatorname*{argmax}_{J,I} \{f(J, I) : |I| = p^{\ell-r}, I \subseteq \tilde{S}_r, |J| = q^{\ell-r}, J \subseteq \tilde{S}_\ell^C\}$.
12: **end if**
13: **end for**
14: $r_0, \ell_0 = \operatorname{argmax}_{r,\ell} f(J_{r,\ell}, I_{r,\ell})$
15: **if** $f_{r_0,\ell_0} > 0$ **then**
16: $\tilde{S} = \tilde{S} \cup J_{r_0,\ell_0} \backslash I_{r_0,\ell_0}$.
17: $\boldsymbol{X}_{\tilde{S}} = \boldsymbol{X}_{\tilde{S}} + \sum_{j \in J_{r_0,\ell_0}} \boldsymbol{v}_j \boldsymbol{v}_j^\top - \sum_{i \in I_{r_0,\ell_0}} \boldsymbol{v}_i \boldsymbol{v}_i^\top$.
18: **for** $j \in J_{r_0,\ell_0}$ **do**
19: $\boldsymbol{\Lambda} = \boldsymbol{\Lambda} - \frac{\boldsymbol{\Lambda} \boldsymbol{v}_j \boldsymbol{v}_j^\top \boldsymbol{\Lambda}}{1 + \boldsymbol{v}_j^\top \boldsymbol{\Lambda} \boldsymbol{v}_j}$
20: **end for**
21: **for** $i \in I_{r_0,\ell_0}$ **do**
22: $\boldsymbol{\Lambda} = \boldsymbol{\Lambda} + \frac{\boldsymbol{\Lambda} \boldsymbol{v}_i \boldsymbol{v}_i^\top \boldsymbol{\Lambda}}{1 - \boldsymbol{v}_i^\top \boldsymbol{\Lambda} \boldsymbol{v}_i}$
23: **end for**
24: Update $\tau_j = \boldsymbol{v}_j^\top \boldsymbol{\Lambda} \boldsymbol{v}_j$ for all $j \in [n]$.
25: Return to line 5.
26: **end if**
27: **return** $\tilde{S}, \boldsymbol{X}_{\tilde{S}}$

B Proofs

In the following, we give complete proofs for all results that were stated in the main text.

B.1 Proofs from Section 3

Proposition 1. In each iteration of Algorithm 1, the following statements hold:

(i) the determinant $\det(\boldsymbol{X}_{\tilde{S}})$ is monotonically strictly increasing from the previous iteration;

(ii) the inverse $\boldsymbol{X}_{\tilde{S}}^{-1}$ is always well-defined;

(iii) the algorithm always terminates in a finite number of iterations.

Proof of Proposition 1. At line 5, for any pair $r, \ell \in M$, suppose that $I_r = \{i_1, \ldots, i_{p^{m-r}q^r}\}$. If $f_{r,\ell} > 0$ at line 9, we may apply Lemma 1 with $\boldsymbol{A} = \sum_{k=1}^{p^{m-r}q^r-1} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top$, $\boldsymbol{B} = \boldsymbol{v}_{i_{p^{m-r}q^r}}\boldsymbol{v}_{i_{p^{m-r}q^r}}^\top$, $\boldsymbol{C} = \boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r-1} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top$, yielding

$$\begin{aligned}
&\det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}\right) + \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \\
&\geq \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r-1} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) + \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}^\top - \boldsymbol{v}_{i_{p^{m-r}q^r}}\boldsymbol{v}_{i_{p^{m-r}q^r}}^\top\right).
\end{aligned}$$

Subtracting $\det(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell})$ from both sides yields

$$\begin{aligned}
&\det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \geq \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r-1} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \\
&+ \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}^\top - \boldsymbol{v}_{i_{p^{m-r}q^r}}\boldsymbol{v}_{i_{p^{m-r}q^r}}^\top\right) - \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}\right) \qquad (11)
\end{aligned}$$

Similarly, for any $t = 1, \ldots, p^{m-r}q^r - 2$, we have

$$\begin{aligned}
&\det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r-t} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \geq \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r-t-1} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \\
&+ \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}^\top - \boldsymbol{v}_{i_{p^{m-r}q^r-t}}\boldsymbol{v}_{i_{p^{m-r}q^r-t}}^\top\right) - \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^\ell \boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}\right) \qquad (12)
\end{aligned}$$

Summing up inequalities (11) and (12) over $t = 1, \ldots, p^{m-r}q^r - 2$, we have

$$
\begin{aligned}
&\det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^{\ell}\boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^{\top}\right) \\
&\geq \sum_{k=1}^{p^{m-r}q^r} \det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^{\ell}\boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^{\top}\right) - \left(p^{m-r}q^r - 1\right)\det\left(\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^{\ell}\boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell}\right) \\
&= \det\left(\boldsymbol{X}_{\tilde{S}}\right)\left(\sum_{k=1}^{p^{m-r}q^r}\left(1 + p^{m-\ell}q^{\ell}\tau_{j_\ell} - \tau_{i_k} - p^{m-\ell}q^{\ell}\tau_{i_k}\tau_{j_\ell} + p^{m-\ell}q^{\ell}\tau_{i_k j_\ell}^2\right)\right. \\
&\quad \left. - \left(p^{m-r}q^r - 1\right)\left(1 + p^{m-\ell}q^{\ell}\tau_{j_\ell}\right)\right) \\
&= \det\left(\boldsymbol{X}_{\tilde{S}}\right)\left(1 + p^{m-\ell}q^{\ell}\tau_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r}\tau_{i_k} - p^{m-\ell}q^{\ell}\tau_{j_\ell}\sum_{k=1}^{p^{m-r}q^r}\tau_{i_k} + p^{m-\ell}q^{\ell}\sum_{k=1}^{p^{m-r}q^r}\tau_{i_k j_\ell}^2\right) \\
&> \det\left(\boldsymbol{X}_{\tilde{S}}\right),
\end{aligned}
$$

where the first equality is due to Lemma 2(iii), the second equality follows by simple calculation, and the last inequality holds because $f(J_\ell, I_r) > 0$. Since in the next iteration, we replace $\boldsymbol{X}_{\tilde{S}}$ by $\boldsymbol{X}_{\tilde{S}} + p^{m-\ell}q^{\ell}\boldsymbol{v}_{j_\ell}\boldsymbol{v}_{j_\ell} - \sum_{k=1}^{p^{m-r}q^r}\boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^{\top}$, we conclude that $\det(\boldsymbol{X}_{\tilde{S}})$ strictly increases in each new iteration. Since we start from a nonsingular matrix and $\det(\boldsymbol{X}_{\tilde{S}})$ is increasing, we have a positive $\det(\boldsymbol{X}_{\tilde{S}})$ value throughout the algorithm. Hence, $\boldsymbol{X}_{\tilde{S}}$ is always nonsingular and X_S^{-1} always exists.

Lastly, note that $\det(\boldsymbol{X}_{\tilde{S}})$ is bounded above by $\det(\sum_{i\in[n]}\boldsymbol{v}_i\boldsymbol{v}_i^{\top})$ and can have a finite number of possible values since there are at most n^B possible choices of multi-set $\tilde{S}$. Therefore, the algorithm terminates in a finite number of iterations. □

Proposition 2. The time complexity of each interchange of the ordinary search (i.e., running lines 2-18 once) in Algorithm 1 is $O(Bn(d^2 + m\log(p)))$.

Proof. Each interchange includes: (i) solving the maximization problem in line 5, (ii) updating the matrix $\boldsymbol{\Lambda}$, and (iii) updating the τ terms.

(i) Let us first consider the maximization problem in line 5. For each $r \in M$ and $j \in [n]$, we have

$$
J_\ell, I_r = \underset{J,I}{\operatorname{argmax}}\left\{ f(J,I) : |I| = p^{m-r}q^r, I \subseteq \tilde{S}_r, c_j = a^\ell, J = \{\underbrace{j, \cdots, j}_{p^{m-\ell}q^\ell \text{ copies}}\}\right\}. \tag{13}
$$

The objective function can be rewritten as

$$\begin{aligned} f(J,I) =& p^{m-\ell}q^{\ell}\tau_j - \sum_{i\in I}\tau_i - p^{m-\ell}q^{\ell}\tau_j\sum_{i\in I}\tau_i + p^{m-\ell}q^{\ell}\sum_{i\in I}\tau_{ij}^2 \\ =& \sum_{i\in I}\left(p^{r-\ell}q^{\ell-r}\tau_j - \tau_i - p^{m-\ell}q^{\ell}\tau_i\tau_j + p^{m-\ell}q^{\ell}\tau_{ij}^2\right). \end{aligned}$$

For fixed $j \in [n]$ with $c_j = a^\ell$, (13) is a simple linear binary program subject to a cardinality constraint, and therefore is equivalent to finding the subset of the largest $p^{m-r}q^r$ values in the sequence $\{p^{r-\ell}q^{\ell-r}\tau_j - \tau_i - p^{m-\ell}q^{\ell}\tau_i\tau_j + p^{m-\ell}q^{\ell}\tau_{ij}^2\}_{i\in\tilde{S}_r}$. Solving this problem requires sorting with time complexity

$$O(|\tilde{S}_r|\log(p^{m-r}q^r)) = O(m|\tilde{S}_r|\log(p)),$$

where the equality is due to $1 \le q < p$. Since we need to solve (13) for each $r \in M$ and $j \in [n]$, the total complexity is

$$O\left(n\sum_{r\in M} m|\tilde{S}_r|\log(p)\right) = O\left(mn|\tilde{S}|\log(p)\right) = O\left(Bmn\log(p)\right),$$

where the second equality follows because $|\tilde{S}| \le p^m < B$.

(ii) When updating the matrix $\mathbf{\Lambda}$, we apply the rank-one update result of Lemma 2(i) or 2(ii) for a total of $|I_{r_0,\ell_0}|+1$ times. The complexity of each rank-one update is $O(d^2)$. Thus, the total complexity of updating $\mathbf{\Lambda}$ is $O(Bd^2)$ since $|I_{r_0}| \le B$.

(iii) We need to compute τ_j for all $j \in [n]$ and τ_{ij} for all $j \in [n]$, $i \in \tilde{S}$. Thus, at most $O(Bn)$ many τ terms are computed at each iteration. Each τ term incurs complexity $O(d^2)$. Thus, the total complexity of updating τ terms is $O(Bnd^2)$.

In conclusion, the total complexity of each interchange is

$$O\left(Bnd^2 + Bnm\log(p) + Bd^2\right) = O\left(Bn\left(d^2 + m\log(p)\right)\right),$$

which completes the proof. □

Lemma 3. After line 18 of Algorithm 1, for any $j \in [n]$, we must have

$$\frac{\tau_j}{c_j} \leq \frac{d}{c(\tilde{S}) - p^m d + p^m}.$$

Proof of Lemma 3. Let $\tilde{S}$ be the set returned when Algorithm 1 terminates. Define $\tilde{S}_r := \{i \in \tilde{S} : c_i = a^r\}$, and let $\tilde{S}_\ell^C = \{j \in [n] : c_j = a^\ell\}$. Take any $r, \ell \in M$ and any $j \in \tilde{S}_\ell^C$, and consider a set $I_r := \{i_1, i_2, \ldots, i_{p^{m-r}q^r}\}$ containing $p^{m-r}q^r$ distinct elements in the set $\tilde{S}_r$. Then, we have

$$f(J_{r,\ell}, I_{r,\ell}) = p^{m-\ell}q^\ell \tau_j - \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k} - p^{m-\ell}q^\ell \tau_j \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k} + p^{m-\ell}q^\ell \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k j}^2 \leq 0, \tag{14}$$

where J_ℓ is the multi-set of $p^{m-\ell}q^\ell$ copies of element j.

Summing over all the possible tuples of $(i_1, i_2, \ldots, i_{p^{m-r}q^r})$ in (14), we have

$$\begin{aligned}
&|\tilde{S}_r|\left(|\tilde{S}_r| - 1\right)\cdots\left(|\tilde{S}_r| - p^{m-r}q^r + 1\right)\cdot p^{m-\ell}q^\ell\tau_j \\
&- p^{m-r}q^r\left(|\tilde{S}_r| - 1\right)\cdots\left(|\tilde{S}_r| - p^{m-r}q^r + 1\right)\sum_{i\in\tilde{S}_r}\tau_i \\
&- p^{m-\ell}q^\ell\tau_j p^{m-r}q^r\left(|\tilde{S}_r| - 1\right)\cdots\left(|\tilde{S}_r| - p^{m-r}q^r + 1\right)\sum_{i\in\tilde{S}_r}\tau_i \\
&+ p^{m-\ell}q^\ell\left(|\tilde{S}_r| - 1\right)\cdots\left(|\tilde{S}_r| - p^{m-r}q^r + 1\right)\cdot p^{m-r}q^r\sum_{i\in\tilde{S}_r}\tau_{ij}^2 \leq 0,
\end{aligned}$$

which is equivalent to

$$|\tilde{S}_r|p^{m-\ell}q^\ell\tau_j - p^{m-r}q^r\sum_{i\in\tilde{S}_r}\tau_i - p^{m-\ell}q^\ell p^{m-r}q^r\tau_j\sum_{i\in\tilde{S}_r}\tau_i + p^{m-\ell}q^\ell p^{m-r}q^r\sum_{i\in\tilde{S}_r}\tau_{ij}^2 \leq 0. \tag{15}$$

Multiplying (15) by a^r and summing them over $r \in M$, we obtain

$$p^{m-\ell}q^\ell\sum_{r\in M}a^r|\tilde{S}_r|\tau_j - p^m\sum_{r\in M}\sum_{i\in\tilde{S}_r}\tau_i - p^{2m-\ell}\tau_j\sum_{r\in M}\sum_{i\in\tilde{S}_r}\tau_i + p^{2m-\ell}\sum_{r\in M}\sum_{i\in\tilde{S}_r}\tau_{ij}^2 \leq 0, \tag{16}$$

Note that the sets $\{\tilde{S}_r\}_{r\in M}$ are disjoint and their union is $\tilde{S} = \bigcup_{r\in M}\tilde{S}_r$ with total cost $\sum_{r\in M}a^r|\tilde{S}_r| =$

$c(\tilde{S})$. Therefore, (16) becomes

$$p^{m-\ell}q^{\ell}c(\tilde{S})\tau_j - p^m\sum_{i\in\tilde{S}}\tau_i - p^{2m-\ell}\tau_j\sum_{i\in\tilde{S}}\tau_i + p^{2m-\ell}\sum_{i\in\tilde{S}}\tau_{ij}^2 \leq 0. \tag{17}$$

By Lemma 2(iv) and (v), we can rewrite (17) as

$$p^{m-\ell}q^{\ell}c(\tilde{S})\tau_j - p^m d - p^{2m-\ell}\tau_j d + p^{2m-\ell}\tau_j \leq 0,$$

or, equivalently,

$$p^m\left(\left(p^{-\ell}q^{\ell}c(\tilde{S}) - p^{m-\ell}\left(d-1\right)\right)\tau_j - d\right) \leq 0.$$

Since $c_j = a^{\ell} = \frac{p^{\ell}}{q^{\ell}}$, we obtain

$$\frac{\tau_j}{c_j} = \frac{\tau_j}{a^{\ell}} \leq \frac{d}{c(\tilde{S}) - \frac{p^m}{q^{\ell}}\left(d-1\right)} \leq \frac{d}{c(\tilde{S}) - p^m d + p^m},$$

where the second inequality is due to $q^l \geq 1$ and $d \geq 1$. □

Theorem 1. Let $\tilde{S}$ be the output of Algorithm 1. Then we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}\left(1 - \frac{\left(\frac{B}{p^m} - \left\lfloor\frac{B}{p^m}\right\rfloor + d - 1\right)p^m}{B}\right) \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}\left(1 - \frac{p^m d}{B}\right).$$

Particularly, when $B > \frac{p^m d}{\epsilon}$ for some $\epsilon \in (0,1)$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq (1-\epsilon)\left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

Proof. Since (REL-W) is a concave program, we may straightforwardly derive the dual

$$\begin{aligned} \min_{\boldsymbol{Y},\mu} \quad & \frac{1}{d}\log\det\left(\boldsymbol{Y}\right) + \frac{B}{d}\mu - 1 \\ \text{s.t.} \quad & -\boldsymbol{v}_i^{\top}\boldsymbol{Y}^{-1}\boldsymbol{v}_i + c_i\mu \geq 0 \quad \forall i \in [n] \\ & \boldsymbol{Y} \succeq 0 \\ & \mu \geq 0. \end{aligned} \tag{DUAL-W}$$

Let $\boldsymbol{Y} = \alpha \boldsymbol{X}_{\tilde{S}}$ for some $\alpha > 0$ and $\mu = \frac{1}{\alpha} \frac{d}{c(\tilde{S}) - p^m d + p^m}$. By Lemma 3, we have

$$-\boldsymbol{v}_i^\top \boldsymbol{Y}^{-1} \boldsymbol{v}_i + c_i \mu = -\frac{1}{\alpha} \tau_i + c_i \mu \geq -c_i \frac{1}{\alpha} \frac{d}{c(\tilde{S}) - p^m d + p^m} + c_i \frac{1}{\alpha} \frac{d}{c(\tilde{S}) - p^m d + p^m} = 0$$

for any $i \in [n]$. Therefore, $(\boldsymbol{Y}, \mu)$ is a dual-feasible solution.

Let S^* be an optimal multi-subset to the combinatorial formulation (1) and $\boldsymbol{X}_{S^*} = \sum_{i \in S^*} \boldsymbol{v}_i \boldsymbol{v}_i^\top$. Let $x_i^* = |\{i : i \in S^*\}|$ denote the number of is in the set S^* for each $i \in [n]$. Then $\boldsymbol{x}^*$ is feasible to (REL-W) with objective $\frac{1}{d} \log\det(\boldsymbol{X}_{S^*})$. Hence, according to the weak duality of the convex program, we have

$$\frac{1}{d} \log\det\left(\boldsymbol{X}_{S^*}\right) \leq \frac{1}{d} \log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\alpha\right) + \frac{1}{\alpha} \frac{B}{d} \frac{d}{c(\tilde{S}) - p^m d + p^m} - 1.$$

Minimizing over α, we obtain the best $\alpha^* = \frac{B}{c(\tilde{S}) - p^m d + p^m}$. Plugging it in, we have

$$\frac{1}{d} \log\det\left(\boldsymbol{X}_{S^*}\right) \leq \frac{1}{d} \log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\frac{B}{c(\tilde{S}) - p^m d + p^m}\right). \tag{18}$$

Recall that ordinary search always performs equal-cost interchanges; furthermore, for $\ell \in M$, we add $p^{m-\ell} q^\ell$ experiments with cost a^ℓ. This yields the inequality $\left\lfloor \frac{B}{p^m} \right\rfloor p^m \leq c(\tilde{S}) \leq B$, whence (18) can be further relaxed as

$$\frac{1}{d} \log\det\left(\boldsymbol{X}_{S^*}\right) \leq \frac{1}{d} \log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\frac{B}{\left(\left\lfloor \frac{B}{p^m} \right\rfloor - d + 1\right) p^m}\right).$$

Therefore,

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{\left(\frac{B}{p^m} - \left\lfloor \frac{B}{p^m} \right\rfloor + d - 1\right) p^m}{B}\right) \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{p^m d}{B}\right),$$

where the second inequality is due to $\frac{B}{p^m} - \left\lfloor \frac{B}{p^m} \right\rfloor - 1 < 0$. When $B > \frac{p^m d}{\epsilon}$ for some $\epsilon \in (0, 1)$, we will have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq (1 - \epsilon) \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}},$$

as desired. □

Theorem 2. Let $\tilde{S}$ be the output of Algorithm 1 with $a = \frac{p}{q}$, $c'_i := a^{\lceil \log_a(c_i) \rceil}$, and $m = \lceil \log_a(c_{\max}) \rceil$. When $B \geq p^m d$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(\frac{q}{p} - q^m \frac{c_{\max} d}{B}\right).$$

Particularly, when $B > q^m \frac{c_{\max} d}{\epsilon}$ for some $\epsilon \in (0, \frac{q}{p})$, we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left(\frac{q}{p} - \epsilon\right) \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

Proof of Theorem 2. In the proof of Theorem 1, letting $\boldsymbol{Y} = \alpha \boldsymbol{X}$, $\mu = \frac{1}{\alpha} \frac{ad}{\sum_{j \in \tilde{S}} c'_j - p^m d + p^m}$ and $\alpha = \frac{aB}{\sum_{j \in \tilde{S}} c'_j - p^m d + p^m}$ and following the same derivation, we have

$$\frac{1}{d} \log\det\left(\boldsymbol{X}_{S^*}\right) \leq \frac{1}{d} \log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\frac{aB}{B - p^m d}\right).$$

Therefore,

$$\begin{aligned}
\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} &\geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(\frac{q}{p} - \frac{p^{m-1} q d}{B}\right) && (19) \\
&\geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(\frac{q}{p} - q^{\left\lceil \log_{\frac{p}{q}}(c_{\max}) \right\rceil} \frac{c_{\max} d}{B}\right) && (20)
\end{aligned}$$

where (19) holds because $m = \lceil \log_a(c_{\max}) \rceil < 1 + \log_a(c_{\max})$ and $(\frac{p}{q})^{m-1} \leq c_{\max}$.

When $B > q^{\lceil \log_{\frac{p}{q}}(c_{\max}) \rceil} \frac{c_{\max} d}{\epsilon}$ for some $\epsilon > 0$, (20) becomes

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left(\frac{q}{p} - \epsilon\right) \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}}.$$

□

B.2 Proofs from Section 4

Lemma 4. For any two index subsequences with distinct elements $I := \{i_1, i_2, \ldots, i_{p^{m-r}q^r}\} \subseteq \tilde{S}$ and $J := \{j_1, j_2, \ldots, j_{p^{m-r}q^r}\} \subseteq \tilde{S}^C := [n] \setminus \tilde{S}$, we have

$$\det\left(\boldsymbol{X}_{\tilde{S}} + \sum_{k=1}^{p^{m-\ell}q^\ell} \boldsymbol{v}_{j_k}\boldsymbol{v}_{j_k}^\top - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \geq \det\left(\boldsymbol{X}_{\tilde{S}}\right)\left(1 + \sum_{k=1}^{p^{m-\ell}q^\ell} \tau_{j_k}\right)\left(1 - \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k}\right). \quad (21)$$

Proof of Lemma 4. Without loss of generality, we can assume that $(1-\sum_{k=1}^{p^{m-r}q^r} \tau_{i_k}) > 0$; otherwise, the inequality (21) is trivial.

Let us first prove $\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top \succ 0$. Similarly to the proof of Proposition 1, applying strong superadditivity yields

$$\begin{aligned}
\det\left(\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) &\geq \sum_{k=1}^{p^{m-r}q^r} \det\left(\boldsymbol{X}_{\tilde{S}} - \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) - \left(p^{m-\ell}q^\ell - 1\right)\det\left(\boldsymbol{X}_{\tilde{S}}\right) \\
&= \det\left(\boldsymbol{X}_{\tilde{S}}\right)\left(1 - \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k}\right) \\
&> 0.
\end{aligned} \quad (22)$$

where the last inequality is due to $\det\left(\boldsymbol{X}_{\tilde{S}}\right)$ and the assumption that $(1 - \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k}) > 0$.

To prove the main result, we write

$$\begin{aligned}
&\det\left(\boldsymbol{X}_{\tilde{S}} + \sum_{k=1}^{p^{m-\ell}q^\ell} \boldsymbol{v}_{j_k}\boldsymbol{v}_{j_k}^\top - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \\
&\geq \sum_{t=1}^{p^{m-\ell}q^\ell} \det\left(\boldsymbol{X}_{\tilde{S}} + \boldsymbol{v}_{j_t}\boldsymbol{v}_{j_t}^\top - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) - \left(p^{m-\ell}q^\ell - 1\right)\det\left(\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right) \\
&= \det\left(\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right)\left(1 + \sum_{t=1}^{p^{m-\ell}q^\ell} \boldsymbol{v}_{j_t}^\top\left(\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right)^{-1}\boldsymbol{v}_{j_t}\right) \\
&\geq \det\left(\boldsymbol{X}_{\tilde{S}}\right)\left(1 - \sum_{k=1}^{p^{m-r}q^r} \tau_{i_k}\right)\left(1 + \sum_{t=1}^{p^{m-\ell}q^\ell} \boldsymbol{v}_{j_t}^\top\left(\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top\right)^{-1}\boldsymbol{v}_{j_t}\right),
\end{aligned}$$

where the first inequality is by strong superadditivity and the similar proof of Theorem 1, the

equality is by the rank-one update in Lemma 2(iii), the last inequality is obtained due to inequality (22) above.

Since $\boldsymbol{X}_{\tilde{S}} \succeq \boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top \succ 0$, we have

$$1 + \sum_{t=1}^{p^{m-\ell}q^\ell} \boldsymbol{v}_{j_t}^\top \left(\boldsymbol{X}_{\tilde{S}} - \sum_{k=1}^{p^{m-r}q^r} \boldsymbol{v}_{i_k}\boldsymbol{v}_{i_k}^\top \right)^{-1} \boldsymbol{v}_{j_t} \geq 1 + \sum_{t=1}^{p^{m-\ell}q^\ell} \boldsymbol{v}_{j_t}^\top \boldsymbol{X}_{\tilde{S}}^{-1} \boldsymbol{v}_{j_t} = 1 + \sum_{t=1}^{p^{m-\ell}q^\ell} \tau_{j_t},$$

which completes the proof. □

Proposition 4. The following inequalities must hold: (i) $\omega \leq \frac{d}{c(\tilde{S})}$, and (ii) $1 - p^m\omega > 0$.

Proof of Proposition 4. For any $r \in M$, we have

$$\omega \leq \frac{1}{p^m} \min_{I \subseteq \tilde{S}_r, |I| = p^{m-r}q^r} \sum_{i \in I} \tau_i = \frac{1}{a^r} \min_{I \subseteq \tilde{S}_r, |I| = p^{m-r}q^r} \frac{\sum_{i \in I} \tau_i}{|I|} \leq \frac{1}{a^r} \frac{\sum_{i \in \tilde{S}_r} \tau_i}{\left|\tilde{S}_r\right|} = \frac{\sum_{i \in \tilde{S}_r} \tau_i}{c(\tilde{S}_r)},$$

where the first inequality follows from the definition of ω, the first equality uses the fact that $|I| = p^{m-r}q^r$, the second inequality holds because larger τ_i values are added to the average, and the last equality holds because $c_i = a^r$ for any $i \in \tilde{S}_r$ and $\sum_{i \in \tilde{S}_r} c_i = c(\tilde{S}_r)$.

Therefore,

$$\omega \leq \min_{r \in M} \frac{\sum_{i \in \tilde{S}_r} \tau_i}{c(\tilde{S}_r)} \leq \frac{\sum_{r \in M} \sum_{i \in \tilde{S}_r} \tau_i}{\sum_{r \in M} c(\tilde{S}_r)} = \frac{\sum_{i \in \tilde{S}} \tau_i}{c(\tilde{S})} = \frac{d}{c(\tilde{S})},$$

where the second inequality holds because the ratio of the average of numerators over the average of denominators is no smaller than the smallest ratios, the last inequality uses Lemma 2(iv). This completes the proof of part (i).

For part (ii), since the budget is large enough such as $B \geq p^m(d + m + 1)$, we have $c(\tilde{S}) \geq \lfloor \frac{B}{p^m} \rfloor p^m \geq p^m(d + m + 1)$. Therefore, by part (i), we have $p^m\omega \leq p^m \frac{d}{c(\tilde{S})} \leq \frac{d}{d+m+1}$, which implies that $1 - p^m\omega > 0$. □

Proposition 5. For each $\ell \in M$, the following statements hold: (i) $|O_\ell^J| \leq p^{m-\ell}q^\ell - 1$ and $|O_\ell^I| \leq p^{m-\ell}q^\ell - 1$; and (ii) $|O_\ell^J| \times |O_\ell^I| = 0$.

Proof of Proposition 5. Part (i) can be proved by contradiction. Suppose that there exist $p^{m-\ell}q^\ell$ distinct elements $j_1, \ldots, j_{p^{m-\ell}q^\ell} \in O_\ell^J$. Then, by the definition of O_ℓ^J, we have $\tau_{j_k} > a^\ell \frac{\omega}{1-p^m\omega}$ for

any $k = 1, 2, \ldots, p^{m-\ell}q^{\ell}$. Summing up all the inequalities, we have

$$\frac{\sum_{k=1}^{p^{m-\ell}q^{\ell}} \tau_{j_k}}{p^m} > \frac{\omega}{1 - p^m\omega},$$

a contradiction to (7).

Similarly, suppose that there exist $p^{m-\ell}q^{\ell}$ distinct elements $i_1, \ldots, i_{p^{m-\ell}q^{\ell}} \in O_\ell^I$. Then we have $\tau_{i_k} < a^{\ell}\omega$ for any $k = 1, 2, \ldots, p^{m-\ell}q^{\ell}$. Summing up all the inequalities, we have

$$\frac{\sum_{k=1}^{p^{m-\ell}q^{\ell}} \tau_{i_k}}{p^m} < \omega,$$

a contradiction to the definition of ω.

To show part (ii), suppose that $O_\ell^J \neq \emptyset$. Then for any $j \in O_\ell^J$ and $i \in O_\ell^I$, by rearranging (9), we have

$$\tau_i \geq \frac{\tau_j}{1 + \tau_j} > \frac{a^{\ell}\frac{\omega}{1-p^m\omega}}{1 + a^{\ell}\frac{\omega}{1-p^m\omega}} = \frac{c_i\omega}{1 - (p^m - a^{\ell})\,\omega} \geq c_i\omega,$$

where the second inequality holds because $\frac{\tau_j}{c_j} = \frac{\tau_j}{a^{\ell}} > \frac{\omega}{1-p^m\omega}$ and the function $\frac{x}{1+x}$ is increasing in x; the equality is due to $c_i = a^{\ell}$; and the last inequality holds because $a^{\ell} \leq p^m$ and $1 - p^m\omega > 0$ (see Proposition 4). This contradicts the definition of O_ℓ^I.

If $O_\ell^I \neq \emptyset$, suppose that $i \in O_\ell^I$. Then, for any $j \in O_\ell^J$, by rearranging (9), we have

$$\tau_j \leq \frac{\tau_i}{1 - \tau_i} < \frac{a^{\ell}\omega}{1 - a^{\ell}\omega} = \frac{c_j\omega}{1 - a^{\ell}\omega} \leq c_j\frac{\omega}{1 - p^m\omega},$$

where the second inequality holds because $\frac{\tau_i}{c_i} < \omega$, $c_i = a^{\ell}$ and the function $\frac{x}{1-x}$ is increasing in x; the equality is due to $c_j = a^{\ell}$; and the last inequality is because $a^{\ell} \leq p^m$ and $1 - p^m\omega > 0$ (see Proposition 4). This contradicts the definition of O_ℓ^J. □

Theorem 3. Let $\tilde{S}$ be the output of Algorithm 2. Then we have

$$\left[\det\left(\boldsymbol{X}_{\tilde{S}}\right)\right]^{\frac{1}{d}} \geq \left[\det\left(\boldsymbol{X}_{S^*}\right)\right]^{\frac{1}{d}} \left(1 - \frac{(d+m+1)\,p^m - \sum_{r=0}^{m-1} a^r}{B + \left(mp^m - \sum_{r=0}^{m-1} a^r\right)}\right).$$

Particularly, when $B \geq \epsilon^{-1}((d+m+1)p^m - \sum_{r=0}^{m-1} a^r) - (mp^m - \sum_{r=0}^{m-1} a^r)$ for some $\epsilon \in (0,1)$, we

have

$$[\det\left(\boldsymbol{X}_{\tilde{S}}\right)]^{\frac{1}{d}} \geq (1-\epsilon)\,[\det\left(\boldsymbol{X}_{S^*}\right)]^{\frac{1}{d}}.$$

Proof of Theorem 3. The proof proceeds in six major steps: 1) relaxing (MIP-WO) and taking its dual, 2) constructing a dual solution, 3) verifying its feasibility, 4) applying weak duality to get a performance bound, 5) using properties of the outlier sets to make the bound more tractable, 6) deriving the final approximation guarantee.

Step 1: Formulating the dual. We take the logarithm of the objective function of (MIP-WO) and consider the continuous relaxation

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & \frac{1}{d}\log\det\left(\sum_{i\in[n]} x_i \boldsymbol{v}_i \boldsymbol{v}_i^\top\right) \\ \text{s.t.} \quad & \sum_{i\in[n]} c_i x_i \leq B \\ & 0 \leq x_i \leq 1 \quad \forall i \in [n]. \end{aligned} \tag{REL-WO}$$

It is then fairly straightforward to derive the dual

$$\begin{aligned} \min_{\boldsymbol{Y},\mu,\boldsymbol{\eta}} \quad & \frac{1}{d}\log\det\left(\boldsymbol{Y}\right) + \frac{B}{d}\mu + \frac{1}{d}\sum_{j\in[n]}\eta_j - 1 \\ \text{s.t.} \quad & -\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j \geq 0 \quad \forall j \in [n] \\ & \boldsymbol{Y} \succeq 0, \mu \geq 0, \eta_j \geq 0 \quad \forall j \in [n]. \end{aligned} \tag{DUAL-WO}$$

Step 2: Constructing a dual solution. Let $\boldsymbol{Y} = \alpha \boldsymbol{X}_{\tilde{S}}$ for some $\alpha > 0$, and let $\mu = \frac{1}{\alpha}\frac{\omega}{1-p^m\omega}$. For any $j \in [n]$ such that $c_j = a^\ell$ for some $\ell \in M$, let

$$\eta_j = \begin{cases} 0 & \text{if } j \in S_\ell^C, j \notin O_\ell^J, \\ \frac{1}{\alpha}\tau_j - c_j\mu & \text{if } j \in O_\ell^J, \\ 0 & \text{if } j \in O_\ell^I, \\ \frac{1}{\alpha}\left(\tau_j - c_j\omega\right) & \text{if } j \in S_\ell, j \notin O_\ell^I, \end{cases}$$

where $\tau_j/\alpha = \boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j$. We will use $(\boldsymbol{Y}, \mu, \boldsymbol{\eta})$ as the dual solution.

Step 3: Verifying dual-feasibility. According to Algorithm 2, we have $\boldsymbol{Y} = \alpha \boldsymbol{X}_{\tilde{S}} \succ \boldsymbol{0}$. By Proposition 4, we have $\mu \geq 0$. By the definition of O_ℓ^J and O_ℓ^I, we also have $\eta_j \geq 0$ for all $j \in [n]$.

It remains to check the constraint $-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j \geq 0$ for any $j \in [n]$. For any $j \in [n]$ such that $c_j = a^\ell$ for some $\ell \in M$, there are four cases to be considered:

(1) If $j \in \tilde{S}_\ell^C$ and $j \notin O_\ell^J$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j = -\frac{1}{\alpha}\tau_j + c_j\mu + \eta_j = -\frac{1}{\alpha}\left(\tau_j - c_j\frac{\omega}{1-p^m\omega}\right) \geq 0,$$

where the inequality is due to the definition of O_ℓ^J.

(2) If $j \in O_\ell^J$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j = 0.$$

(3) If $j \in O_\ell^I$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j = \frac{1}{\alpha}\left(-\tau_j + c_j\frac{\omega}{1-p^m\omega}\right) > \frac{1}{\alpha}c_j\left(-\omega + \frac{\omega}{1-p^m\omega}\right) = \frac{1}{\alpha}c_j\frac{p^m\omega^2}{1-p^m\omega} \geq 0.$$

where the first inequality follows from the definition of O_ℓ^I, and the last inequality holds because $\omega \leq \frac{d}{c(S)} < \frac{1}{p^m}$ by Proposition 4.

(4) If $j \in \tilde{S}_\ell$ and $j \notin O_\ell^I$, then

$$\begin{aligned}-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j &= \frac{1}{\alpha}\left(-\tau_j + c_j\frac{\omega}{1-p^m\omega} + \tau_j - c_j\omega\right) = \frac{1}{\alpha}c_j\left(\frac{\omega}{1-p^m\omega} - \omega\right) \\ &= \frac{1}{\alpha}c_j\frac{p^m\omega^2}{1-p^m\omega} \geq 0\end{aligned}$$

where the inequality is because $\omega \leq \frac{d}{c(S)} < \frac{1}{p^m}$ by Proposition 4.

Therefore, $(\boldsymbol{Y}, \mu, \boldsymbol{\eta})$ is a dual feasible solution.

Step 4: Applying weak duality. Let S^* be an optimal subset to the combinatorial formulation (1) and $\boldsymbol{X}_{S^*} = \sum_{i\in S^*} \boldsymbol{v}_i\boldsymbol{v}_i^\top$. For each $i \in [n]$, let $x_i^* = 1$ if $i \in S^*$ and $x_i^* = 0$ otherwise. Then, $\boldsymbol{x}^*$ is a feasible solution of (REL-WO) with objective value $\frac{1}{d}\log\det(\boldsymbol{X}_{S^*})$. Therefore, by weak duality, we

have

$$\begin{aligned}
\frac{1}{d}\log\det\left(\boldsymbol{X}_{S^*}\right) \leq & \frac{1}{d}\log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\alpha\right) + \frac{1}{\alpha}\frac{B}{d}\frac{\omega}{1-p^m\omega} + \frac{1}{\alpha d}\sum_{\ell\in M}\sum_{j\in O_\ell^J}\left(\tau_j - c_j\frac{\omega}{1-p^m\omega}\right) \\
& + \frac{1}{\alpha d}\sum_{r\in M}\sum_{i\in\tilde{S}_r, i\notin O_r^I}\left(\tau_i - c_i\omega\right) - 1 \\
= & \frac{1}{d}\log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\alpha\right) + \frac{1}{\alpha}\frac{B}{d}\frac{\omega}{1-p^m\omega} + \frac{1}{\alpha d}\sum_{\ell\in M}\sum_{j\in O_\ell^J}\left(\tau_j - c_j\frac{\omega}{1-p^m\omega}\right) \\
& + \frac{1}{\alpha d}\sum_{i\in\tilde{S}}\left(\tau_i - c_i\omega\right) - \frac{1}{\alpha d}\sum_{r\in M}\sum_{i\in O_r^I}\left(\tau_i - c_i\omega\right) - 1 \\
= & \frac{1}{d}\log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\alpha\right) + \frac{1}{\alpha d}\left(\frac{B}{1-p^m\omega} - c(\tilde{S})\right)\omega \\
& + \frac{1}{\alpha d}\sum_{\ell\in M}a^\ell\sum_{j\in O_\ell^J}\left(\frac{\tau_j}{c_j} - \frac{\omega}{1-p^m\omega}\right) + \frac{1}{\alpha} + \frac{1}{\alpha d}\sum_{r\in M}a^r\sum_{i\in O_r^I}\left(\omega - \frac{\tau_i}{c_i}\right) - 1 \qquad (23)
\end{aligned}$$

where the last inequality uses Lemma 2(iv). Minimizing over α, we obtain

$$\alpha^* = \frac{1}{d}\left(\left(\frac{B}{1-p^m\omega} - c(\tilde{S})\right)\omega + \sum_{\ell\in M}a^\ell\sum_{j\in O_\ell^J}\left(\frac{\tau_j}{c_j} - \frac{\omega}{1-p^m\omega}\right) + \sum_{r\in M}a^r\sum_{i\in O_r^I}\left(\omega - \frac{\tau_i}{c_i}\right)\right) + 1. \qquad (24)$$

Plugging this value into (23) yields

$$\frac{1}{d}\log\det\left(X_{\tilde{S}^*}\right) \leq \frac{1}{d}\log\det\left(X_{\tilde{S}}\right) + \log\left(\alpha^*\right) \qquad (25)$$

Our remaining task is to find a tight upper bound of α^*.

Step 5: Making the bound explicit. To derive an upper bound of α^*, let us investigate the terms $\sum_{j\in O_\ell^J}(\frac{\tau_j}{c_j} - \frac{\omega}{1-p^m\omega})$ and $\sum_{i\in O_r^I}(\omega - \frac{\tau_i}{c_i})$ in (24).

Suppose that $O_\ell^J = \{j_1, \ldots, j_{|O_\ell^J|}\}$, and the set $\{j_{|O_\ell^J|+1}, \ldots, j_{p^{m-\ell}q^\ell}\}$ denotes a size-$p^{m-\ell}q^\ell$ subset of $\tilde{S}_\ell^C \backslash O_\ell^J$. By the local optimality condition (7), we have

$$\sum_{j\in O_\ell^J}\tau_j \leq \sum_{k=1}^{p^{m-\ell}q^\ell}\tau_{j_k} \leq p^m\frac{\omega}{1-p^m\omega}, \qquad (26)$$

where the first inequality holds because we add nonnegative terms ($\sum_{k=|O_\ell^J|+1}^{p^{m-\ell}q^\ell} \tau_{j_k}$) to the sum, and the second inequality is due to (7).

Therefore,

$$\sum_{j\in O_\ell^J} \frac{\tau_j}{c_j} = \frac{\sum_{j\in O_\ell^J} \tau_j}{a^\ell} \leq \frac{p^m}{a^\ell} \frac{\omega}{1-p^m\omega} = p^{m-\ell}q^\ell \frac{\omega}{1-p^m\omega}, \tag{27}$$

where the first equality holds because $c_j = a^\ell$ for $j \in O_\ell^J$, the inequality is due to (26), and the last equality holds because $a = p/q$. Consequently, if $|O_\ell^J| \neq 0$, we have

$$\sum_{j\in O_\ell^J} \left(\frac{\tau_j}{c_j} - \frac{\omega}{1-p^m\omega}\right) \leq \left(p^{m-\ell}q^\ell - |O_\ell^J|\right) \frac{\omega}{1-p^m\omega} \leq \left(p^{m-\ell}q^\ell - 1\right) \frac{\omega}{1-p^m\omega}, \tag{28}$$

where the first inequality is due to (27) and the fact that $\frac{\omega}{1-p^m\omega}$ is independent of $|O_\ell^J|$, and the second inequality is because $|O_\ell^J| \geq 1$ holds with equality when $|O_\ell^J| = 1$. Note that if $|O_\ell^J| = 0$, then (28) trivially holds because the left-hand side is 0.

On the other hand,

$$\sum_{i\in O_r^I} \left(\omega - \frac{\tau_i}{c_i}\right) \leq |O_r^I|\omega \leq \left(p^{m-r}q^r - 1\right)\omega, \tag{29}$$

where the first inequality holds because $\frac{\tau_i}{c_i} \geq 0$ for any $i \in O_r^I$ and the second inequality is by Proposition 5.

Note that, in (24), the second term on the right-hand side depends on O_ℓ^J and the third term depends on O_r^I. Using (28) and (29), the upper bound of the second term dominates that of the third term. According to Proposition 5, since O_ℓ^J and O_ℓ^I cannot be simultaneously nonempty for any $\ell \in M$, the worst case is when O_ℓ^I is empty for all $\ell \in M$ and the inequalities in (28) hold with equality. Then, by plugging (28) into (24) and removing the third term, we have

$$\begin{aligned}
\alpha^* &\leq \frac{1}{d}\left(\left(\frac{B}{1-p^m\omega} - c(\tilde{S})\right)\omega + \sum_{\ell=0}^{m-1}\left(\frac{p}{q}\right)^\ell \left(p^{m-\ell}q^\ell - 1\right)\frac{\omega}{1-p^m\omega}\right) + 1 \\
&= \frac{1}{d}\left(\frac{B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{1-p^m\omega} - c(\tilde{S})\right)\omega + 1 \\
&\leq \frac{1}{d}\left(\frac{B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{1-p^m\frac{d}{c(\tilde{S})}} - c(\tilde{S})\right)\frac{d}{c(\tilde{S})} + 1
\end{aligned}$$

$$= \frac{B + mp^m - \sum_{r=0}^{m-1} \left(\frac{p}{q}\right)^r}{c(\tilde{S}) - p^m d}, \tag{30}$$

where the equalities are algebraic computations, the second inequality uses Proposition 4.

Step 6: Deriving the final bound. Using the fact that $c(\tilde{S}) \geq \lfloor \frac{B}{p^m} \rfloor p^m$ and plugging (30) into (25), we have

$$\frac{1}{d} \log \det \left(\boldsymbol{X}_{S^*} \right) \leq \frac{1}{d} \log \det \left(\boldsymbol{X}_{\tilde{S}} \right) + \log \left(\frac{B + mp^m - \sum_{r=0}^{m-1} \left(\frac{p}{q}\right)^r}{\left\lfloor \frac{B}{p^m} \right\rfloor p^m - p^m d} \right),$$

which is equivalent to

$$\begin{aligned} \left[\det \left(X_{\tilde{S}} \right) \right]^{\frac{1}{d}} &\geq \left[\det \left(X_{S^*} \right) \right]^{\frac{1}{d}} \left(1 - \frac{\left(\frac{B}{p^m} - \left\lfloor \frac{B}{p^m} \right\rfloor + d \right) p^m + mp^m - \sum_{r=0}^{m-1} \left(\frac{p}{q}\right)^r}{B + mp^m - \sum_{r=0}^{m-1} \left(\frac{p}{q}\right)^r} \right) \\ &\geq \left[\det \left(X_{S^*} \right) \right]^{\frac{1}{d}} \left(1 - \frac{(1 + d + m) p^m - \sum_{r=0}^{m-1} \left(\frac{p}{q}\right)^r}{B + mp^m - \sum_{r=0}^{m-1} \left(\frac{p}{q}\right)^r} \right), \end{aligned}$$

where the second inequality is due to $\frac{B}{p^m} - \left\lfloor \frac{B}{p^m} \right\rfloor - 1 < 0$.

Notably, when $B \geq \epsilon^{-1}((d + m + 1)p^m - \sum_{r=0}^{m-1}(\frac{p}{q})^r) - (mp^m - \sum_{r=0}^{m-1}(\frac{p}{q})^r)$ for some $\epsilon \in (0, 1)$, we have

$$\left[\det \left(X_{\tilde{S}} \right) \right]^{\frac{1}{d}} \geq (1 - \epsilon) \left[\det \left(X_{S^*} \right) \right]^{\frac{1}{d}},$$

which completes the proof. □

Theorem 4. Let $\tilde{S}$ be the output of Algorithm 2 with $a = \frac{p}{q}$, $c_i' := a^{\lceil \log_a(c_i) \rceil}$, $m = \lceil \log_a(c_{\max}) \rceil$. If $B \geq p^m(d + m + 1)$, then we have

$$\left[\det \left(\boldsymbol{X}_{\tilde{S}} \right) \right]^{\frac{1}{d}} \geq \left[\det \left(\boldsymbol{X}_{S^*} \right) \right]^{\frac{1}{d}} \left(\frac{q}{p} - q^m \frac{c_{\max} (d + m + 1)}{B} \right).$$

Particularly, when $B > q^m \frac{c_{\max}(d+m+1)}{\epsilon}$ for some $\epsilon \in (0, \frac{q}{p})$, we have

$$\left[\det \left(\boldsymbol{X}_{\tilde{S}} \right) \right]^{\frac{1}{d}} \geq \left(\frac{q}{p} - \epsilon \right) \left[\det \left(\boldsymbol{X}_{S^*} \right) \right]^{\frac{1}{d}}.$$

Proof of Theorem 4. For $r, \ell \in M$, let us denote $\tilde{S}_r := \{i \in S : c'_i = a^r\}$, $\tilde{S}^C_\ell := \{j \notin S : c'_j = a^\ell\}$ and define

$$O^J_\ell = \left\{ j \in \tilde{S}^C_\ell : \frac{\tau_j}{c'_j} > \frac{\omega}{1 - p^m \omega} \right\}, O^I_r = \left\{ i \in \tilde{S}_r : \frac{\tau_i}{c'_i} < \omega \right\}.$$

The proof proceeds along the same lines as the proof of Theorem 3, but there are some key differences. The analysis can be divided into three steps.

Step 1: Constructing a dual solution. Let $\boldsymbol{Y} = \alpha \boldsymbol{X}_S$ for some $\alpha > 0$ and $\mu = \frac{1}{\alpha} \frac{p}{q} \frac{\omega}{1 - p^m \omega}$. For any $j \in [n]$ such that $c'_j = a^\ell$,

$$\eta_j = \begin{cases} 0 & \text{if } j \in \tilde{S}^C_\ell, j \notin O^J_\ell \\ \frac{1}{\alpha} \left(\tau_j - c'_j \frac{\omega}{1 - p^m \omega} \right) & \text{if } j \in O^J_\ell \\ 0 & \text{if } j \in O^I_\ell \\ \frac{1}{\alpha} \left(\tau_j - c'_j \omega \right) & \text{if } j \in \tilde{S}_\ell, j \notin O^I_\ell, \end{cases}$$

where $\tau_j / \alpha = \boldsymbol{v}_j^\top \boldsymbol{Y}^{-1} \boldsymbol{v}_j$.

Step 2: Verifying dual-feasibility. Again, the proposed dual solution is $(\boldsymbol{Y}, \mu, \boldsymbol{\eta})$. According to Algorithm 2, we have $\boldsymbol{Y} = \alpha \boldsymbol{X}_{\tilde{S}} \succ \boldsymbol{0}$. According to Proposition 4, we have $\mu \geq 0$. By the definition of O^J_ℓ and O^I_ℓ, we also have $\eta_j \geq 0$ for all $j \in [n]$.

It remains to check the constraint $-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1} \boldsymbol{v}_j + c_j \mu + \eta_j \geq 0$ for any $j \in [n]$. For any $j \in [n]$ such that $c'_j = a^\ell$, there are four cases to be considered:

(1) If $j \in \tilde{S}^C_\ell$ and $j \notin O^J_\ell$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1} \boldsymbol{v}_j + c_j \mu + \eta_j = -\frac{1}{\alpha} \tau_j + c_j \mu + \eta_j = -\frac{1}{\alpha} \left(\tau_j - c_j \frac{p}{q} \frac{\omega}{1 - p^m \omega} \right) \geq 0,$$

where the inequality is from the definition of O^J_ℓ and $1 \leq \frac{c'_j}{c_j} \leq \frac{p}{q}$.

(2) If $j \in O^J_\ell$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1} \boldsymbol{v}_j + c_j \mu + \eta_j = \left(c_j \frac{p}{q} - c'_j \right) \frac{\omega}{1 - p^m \omega} \geq 0.$$

(3) If $j \in O_\ell^I$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j = \frac{1}{\alpha}\left(-\tau_j + c_j\frac{p}{q}\frac{\omega}{1-p^m\omega}\right) > \frac{1}{\alpha}\left(-c_j'\omega + c_j\frac{p}{q}\frac{\omega}{1-p^m\omega}\right)$$
$$\geq \frac{1}{\alpha}\left(-c_j' + c_j\frac{p}{q}\right)\omega \geq 0,$$

where the first inequality is due to the definition of O_ℓ^I, and the second inequality holds because $\omega \leq \frac{d}{c'(S)} < \frac{1}{p^m}$ by Proposition 4.

(4) If $j \in \tilde{S}_\ell$ and $j \notin O_\ell^I$, then

$$-\boldsymbol{v}_j^\top \boldsymbol{Y}^{-1}\boldsymbol{v}_j + c_j\mu + \eta_j = \frac{1}{\alpha}\left(c_j\frac{p}{q}\frac{\omega}{1-p^m\omega} - c_j'\omega\right) \geq \frac{1}{\alpha}c_j'\left(\frac{\omega}{1-p^m\omega} - \omega\right) = \frac{1}{\alpha}c_j'\frac{p^m\omega^2}{1-p^m\omega} \geq 0,$$

where the inequality holds because $\omega \leq \frac{d}{c'(S)} < \frac{1}{p^m}$ by Proposition 4.

Therefore, $(\boldsymbol{Y}, \mu, \boldsymbol{\eta})$ is a dual feasible solution.

Step 3: Completing the proof. Let S^* be an optimal subset to the combinatorial formulation (1) and $\boldsymbol{X}_{S^*} = \sum_{i\in S^*} \boldsymbol{v}_i\boldsymbol{v}_i^\top$. For each $i \in [n]$, let $x_i^* = 1$ if $i \in S^*$ and $x_i^* = 0$ otherwise. Then $\boldsymbol{x}^*$ is a feasible solution of (REL-WO) with objective value $\frac{1}{d}\log\det(\boldsymbol{X}_{S^*})$. Hence, by weak duality, we have

$$\frac{1}{d}\log\det\left(\boldsymbol{X}_{S^*}\right) \leq \frac{1}{d}\log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\alpha\right) + \frac{1}{\alpha d}\left(\frac{p}{q}\frac{B}{1-p^m\omega} - c'\left(\tilde{S}\right)\right)\omega$$
$$+ \frac{1}{\alpha d}\sum_{\ell\in M} a^\ell \sum_{j\in O_\ell^J}\left(\frac{\tau_j}{c_j'} - \frac{\omega}{1-p^m\omega}\right) + \frac{1}{\alpha} + \frac{1}{\alpha d}\sum_{r\in M} a^r \sum_{i\in O_r^I}\left(\omega - \frac{\tau_i}{c_i'}\right) - 1, \quad (31)$$

using Lemma 2(iv) as in the proof of Theorem 3. Minimizing over α, we obtain

$$\alpha^* = \frac{1}{d}\left(\left(\frac{p}{q}\frac{B}{1-p^m\omega} - c'(\tilde{S})\right)\omega + \sum_{\ell\in M} a^\ell \sum_{j\in O_\ell^J}\left(\frac{\tau_j}{c_j'} - \frac{\omega}{1-p^m\omega}\right) + \sum_{r\in M} a^r \sum_{i\in O_r^I}\left(\omega - \frac{\tau_i}{c_i'}\right)\right) + 1. \quad (32)$$

Plugging this value into (31), and using Step V in the proof of Theorem 3, we have

$$\alpha^* \leq \frac{1}{d}\left(\left(\frac{p}{q}\frac{B}{1-p^m\omega} - c'\left(\tilde{S}\right)\right)\omega + \sum_{\ell=0}^{m-1}\left(\frac{p}{q}\right)^\ell\left(p^{m-\ell}q^\ell - 1\right)\frac{\omega}{1-p^m\omega}\right) + 1$$

$$
\begin{aligned}
&= \frac{1}{d}\left(\left(\frac{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{1 - p^m\omega} - c'\left(\tilde{S}\right)\right)\omega\right) + 1 \\
&\leq \frac{1}{d}\left(\left(\frac{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{1 - p^m\frac{d}{c'\left(\tilde{S}\right)}} - c'\left(\tilde{S}\right)\right)\frac{d}{c'\left(\tilde{S}\right)}\right) + 1 \\
&= \frac{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{c'\left(\tilde{S}\right) - p^m d} \\
&\leq \frac{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{\left\lfloor\frac{B}{p^m}\right\rfloor p^m - p^m d},
\end{aligned} \tag{33}
$$

where the second inequality uses Proposition 4, and the last inequality follows because $c'(\tilde{S}) \geq \lfloor\frac{B}{p^m}\rfloor$.

By plugging the formula of α^* into (31) and using (33), we have

$$
\begin{aligned}
\frac{1}{d}\log\det\left(\boldsymbol{X}_{S^*}\right) &\leq \frac{1}{d}\log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\frac{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{\left\lfloor\frac{B}{p^m}\right\rfloor p^m - p^m d}\right) \\
&\leq \frac{1}{d}\log\det\left(\boldsymbol{X}_{\tilde{S}}\right) + \log\left(\frac{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r}{B - p^m(d+1)}\right),
\end{aligned} \tag{34}
$$

where the second inequality is due to $\left\lfloor\frac{B}{p^m}\right\rfloor > \frac{B}{p^m} - 1$.

Rearranging (34) as an approximation ratio in terms of $[\det(\boldsymbol{X}_{\tilde{S}})]^{\frac{1}{d}}$, we have

$$
\begin{aligned}
\frac{[\det(\boldsymbol{X}_{\tilde{S}})]^{\frac{1}{d}}}{[\det(\boldsymbol{X}_{S^*})]^{\frac{1}{d}}} \geq& \frac{B - p^m(d+1)}{\frac{p}{q}B + mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r} \\
\geq& \frac{q}{p}\left(\frac{B - p^m(d+1)}{B + \left(mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r\right)}\right) = \frac{q}{p} - \frac{qp^{m-1}(1+d+m) - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^{r-1}}{B + \left(mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r\right)} \\
\geq& \frac{q}{p} - q^m\frac{c_{\max}(1+d+m) - q^{-m}\sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^{r-1}}{B + \left(mp^m - \sum_{r=0}^{m-1}\left(\frac{p}{q}\right)^r\right)} \\
\geq& \frac{q}{p} - q^m\frac{c_{\max}(1+d+m)}{B}
\end{aligned}
$$

where the second inequality holds because $p/q > 1$, the third inequality follows from the fact that $qp^{m-1} = q^m(\frac{p}{q})^{m-1} \leq q^m c_{\max}$ according to the definition of m, and the last inequality holds because

$(mp^m - \sum_{r=0}^{m-1}(\frac{p}{q})^r) \geq 0$ and $q^{-m}\sum_{r=0}^{m-1}(\frac{p}{q})^{r-1} \geq 0$. □

B.3 Proofs from Section 6

Lemma 8. Let $X_1, ..., X_n$ be independent 0/1 random variables, each with a weight $0 \leq w_1, ..., w_n \leq 1$, and let X be the weighted sum $X = w_1X_1 + \ldots + w_nX_n$. For $\mu = \mathbb{E}[X]$ and every $\bar{\epsilon} > 0$, we have

$$\mathbb{P}\left[X \geq (1+\bar{\epsilon})\,\mu\right] \leq e^{-\frac{\bar{\epsilon}^2}{2+\bar{\epsilon}}\mu}.$$

Proof of Lemma 8. By Corollary 1 in Panconesi & Srinivasan (1997), we have

$$\mathbb{P}\left[X \geq (1+\bar{\epsilon})\,\mu\right] \leq \left(\frac{e^{\bar{\epsilon}}}{(1+\bar{\epsilon})^{1+\bar{\epsilon}}}\right)^{\mu} = e^{\mu(\bar{\epsilon}-(1+\bar{\epsilon})\log(1+\bar{\epsilon}))}.$$

We then apply the inequality $\log(1+x) \geq \frac{2x}{2+x}$ and obtain

$$\mathbb{P}\left[X \geq (1+\bar{\epsilon})\,\mu\right] \leq e^{-\frac{\bar{\epsilon}^2}{2+\bar{\epsilon}}\mu},$$

completing the proof. □

Lemma 7. Let $\tilde{S}$ be a random set obtained from one iteration of the while loop described in lines 3-9) of Algorithm 4. Given $T \subseteq [n]$ with $|T| = d \leq n$ and $\epsilon \in (0,1)$, we have

$$\frac{1}{\prod_{i\in T}\hat{x}_i}\mathbb{P}\left[T \subseteq \tilde{S} | c(\tilde{S}) \leq B\right] \geq (1+\epsilon)^{-d}\left(1 - e^{-\frac{(\epsilon B-(1+\epsilon)c(T))^2}{c_{\max}B(2+\epsilon)(1+\epsilon)}}\right) := \alpha^d.$$

Proof of Lemma 7. We proceed similarly to the proof of Lemma 4 in Singh & Xie (2020), with the main difference being the use of the weighted Chernoff bound. First, we scale the costs $\boldsymbol{c}$ and B by $c_{\max}$, that is, we let $c_i := c_i/c_{\max}$ for each $i \in [n]$ and $B := B/c_{\max}$. After scaling, we have $c_i \leq 1$ and we can apply Lemma 8.

Let X_i be the Bernoulli random variable with success probability

$$\mathbb{P}\{X_i = 1\} = \frac{\hat{x}_i}{1+\epsilon}.$$

Success, in this case, indicates whether $i \in [n]$ is included in $\tilde{S}$, so we have

$$\prod_{i \in T} \hat{x}_i = (1+\epsilon)^d \prod_{i \in T} \mathbb{P}(X_i = 1) = (1+\epsilon)^d \mathbb{P}\{T \subseteq \tilde{S}\}.$$

Combining this with $\mathbb{P}\{c(\tilde{S}) \leq B\} \leq 1$ and plugging these into

$$\frac{1}{\prod_{i \in T} \hat{x}_i} \mathbb{P}\{T \subseteq \tilde{S} | c(\tilde{S}) \leq B\} = \frac{\mathbb{P}\{T \subseteq \tilde{S}, c(\tilde{S}) \leq B\}}{\prod_{i \in T} \hat{x}_i \mathbb{P}\{c(\tilde{S}) \leq B\}},$$

we obtain the lower bound

$$\begin{aligned} \frac{1}{\prod_{i \in T} \hat{x}_i} \mathbb{P}\{T \subseteq \tilde{S} | c(\tilde{S}) \leq B\} &\geq (1+\epsilon)^{-d} \mathbb{P}\{c(\tilde{S}) \leq B | T \subseteq \tilde{S}\} \\ &= (1+\epsilon)^{-d} \mathbb{P}\left\{ \sum_{i \in [n] \setminus T} c_i X_i \leq B - c(T) \right\}. \end{aligned}$$

By Lemma 8, given $\bar{\epsilon} > 0$, we have

$$\mathbb{P}\left[\sum_{i \in [n] \setminus T} c_i X_i > (1+\bar{\epsilon}) \, \mathbb{E}\left[\sum_{i \in [n] \setminus T} c_i X_i \right] \right] \leq e^{-\frac{\bar{\epsilon}^2}{2+\bar{\epsilon}} \frac{\sum_{i \in [n] \setminus T} c_i \hat{x}_i}{1+\epsilon}}.$$

Let $\bar{\epsilon} = \frac{(1+\epsilon)(B - c(T))}{\sum_{i \in [n] \setminus T} c_i \hat{x}_i} - 1$. Then we have

$$\mathbb{P}\left[\sum_{i \in [n] \setminus T} c_i X_i \geq B - c\left(T\right) \right] \leq e^{-\frac{\bar{\epsilon}^2}{2+\bar{\epsilon}} \frac{\sum_{i \in [n] \setminus T} c_i \hat{x}_i}{1+\epsilon}}.$$

Note that

$$\epsilon B - (1+\epsilon) c(T) \leq \bar{\epsilon} \sum_{i \in [n] \setminus T} c_i \hat{x}_i = (1+\epsilon)(B - c(T)) - \sum_{i \in [n] \setminus T} c_i \hat{x}_i \leq \epsilon (B - c(T)),$$

where the first inequality holds because $\sum_{i \in [n] \setminus T} c_i \hat{x}_i \leq \sum_{i \in [n]} c_i \hat{x}_i = B$, and the second inequality holds because

$$\sum_{i \in [n] \setminus T} c_i \hat{x}_i = \sum_{i \in [n]} c_i \hat{x}_i - \sum_{i \in T} c_i \hat{x}_i \geq B - \sum_{i \in T} c_i = B - c(T)$$

and $\sum_{i\in[n]} c_i\hat{x}_i = B$ at optimality. Therefore,

$$\mathbb{P}\left[\sum_{i\in[n]\setminus T} c_i X_i \geq B - c\left(T\right)\right] \leq e^{-\frac{(\epsilon B-(1+\epsilon)c(T))^2}{B(2+\epsilon)(1+\epsilon)}},$$

where we use the fact that $\bar{\epsilon} < \epsilon$.

Now we have

$$(1+\epsilon)^{-d}\,\mathbb{P}\left[\sum_{i\in[n]\setminus T} c_i X_i \leq B - c\left(T\right)\right] \geq (1+\epsilon)^{-d}\left(1 - e^{-\frac{(\epsilon B-(1+\epsilon)c(T))^2}{B(2+\epsilon)(1+\epsilon)}}\right).$$

After scaling back $\boldsymbol{c}$ and B by $c_{\max}$, we have

$$(1+\epsilon)^{-d}\,\mathbb{P}\left[\sum_{i\in[n]\setminus T} c_i X_i \leq B - c\left(T\right)\right] \geq (1+\epsilon)^{-d}\left(1 - e^{-\frac{(\epsilon B-(1+\epsilon)c(T))^2}{c_{\max}B(2+\epsilon)(1+\epsilon)}}\right),$$

which completes the proof. □

Proposition 6. Given $\epsilon \in (0,1)$, suppose that $B > \frac{4c_{\max}d}{\epsilon} + \frac{12c_{\max}}{\epsilon^2}\log(\frac{1}{\epsilon})$. Then the expected number of iterations (lines 3-9) in Algorithm 4 is $O(1)$.

Proof of Proposition 6. By applying Lemma 8 with $\bar{\epsilon} = \epsilon$ and $X = \sum_{i=1}^{n} c_i X_i$, the probability that a sampled solution is feasible is bounded below by

$$\mathbb{P}\left[\sum_{i\in[n]} c_i X_i \leq B\right] \geq 1 - e^{-\frac{\epsilon^2}{(2+\epsilon)(1+\epsilon)}\frac{B}{c_{\max}}}.$$

Therefore, the expected number of iterations is bounded above by the expectation of a geometric distribution with success probability $1 - e^{-\frac{\epsilon^2}{(2+\epsilon)(1+\epsilon)}\frac{B}{c_{\max}}}$. Thus, the expected number of iterations is bounded above by $\frac{1}{1-e^{-\frac{\epsilon^2}{(2+\epsilon)(1+\epsilon)}\frac{B}{c_{\max}}}}$.

Fix any $\epsilon \in (0,1)$. Since $B > \frac{4c_{\max}d}{\epsilon} + \frac{12c_{\max}}{\epsilon^2}\log(\frac{1}{\epsilon})$, we have

$$\frac{\epsilon^2}{(2+\epsilon)(1+\epsilon)}\frac{B}{c_{\max}} > \frac{\epsilon^2}{6}\left(\frac{4d}{\epsilon} + \frac{12}{\epsilon^2}\log\left(\frac{1}{\epsilon}\right)\right) = 2\left(\frac{\epsilon d}{3} - \log\left(\epsilon\right)\right).$$

Let $f(\epsilon) = \frac{\epsilon d}{3} - \log(\epsilon)$. It is straightforward to verify that $f(\epsilon)$ achieves its minimum at $\epsilon = \frac{3}{d}$ if

$d \geq 3$ or at $\epsilon = 1$ if $d \leq 2$. In the former case,

$$f(\epsilon) \geq f\left(\frac{3}{d}\right) \geq 1 - \log\left(\frac{3}{d}\right) \geq 1.$$

In the latter case,

$$f(\epsilon) \geq f(1) \geq \frac{d}{3} \geq \frac{1}{3}.$$

In general $f(\epsilon) \geq \frac{1}{3}$. Therefore, the expected number of iterations is bounded above by $\frac{1}{1-e^{-\frac{2}{3}}}$. $\square$