

Finding a Maximal Determinant Principal Submatrix via Hadamard’s Inequality and Conic Relaxations

Hao Hu* Stefan Sremac† Hugo J. Woerdeman‡ Henry Wolkowicz§

11:21pm, August 18, 2025

Abstract

An important yet challenging problem in numerical linear algebra is finding a principal submatrix with the maximum determinant. In this paper, we examine several exact and approximate approaches to this problem. We first propose an upper bound based on Hadamard’s inequality, along with a projection scheme based on the Gram–Schmidt process without normalization. This scheme yields a highly effective exact algorithm for solving small- to medium-scale instances. We then study a linear programming (LP) relaxation that facilitates reliable performance evaluation when the exact method returns only near-optimal solutions, and prove that our projection scheme also strengthens the upper bound obtained from the LP relaxation. Finally, we present stronger upper bounds via semidefinite programming, further illustrating the intrinsic difficulty of determinant maximization.

Keywords: Semidefinite programming, maximum determinant, maximizing log det, Hadamard’s inequality, conic relaxation.

AMS subject classifications: Primary: 15A15; Secondary: 90C22, 05C85.

1 Introduction

The problem of identifying submatrices with maximum determinant arises naturally in several areas of mathematics including combinatorics, numerical linear algebra, experimental design, machine learning, and statistics. Specifically, given a real symmetric positive semidefinite matrix $M \in \mathbb{R}^{n \times n}$, a principal submatrix of M is obtained by selecting a subset of indices $K \subseteq \{1, \dots, n\}$ and extracting the rows and columns corresponding to K . The task of selecting a subset K of fixed cardinality r such that the resulting *principal submatrix*, M_K , has maximum determinant is known as the *maximum determinant principal submatrix problem*, or *MAXDET*. In numerical linear algebra finding well-conditioned submatrices in large-scale matrices is importance, and selecting a submatrix with large determinant modulus is a useful relaxation. Note that this follows from the fact that $\omega(M) = \frac{\text{trace}(M)/n}{\det(M)^{1/n}}$, the ratio of the arithmetic and geometric means of the eigenvalues, is a valid condition number, see e.g., [17].

This problem is computationally challenging: it is NP-hard in general [7, 19], and its approximation is also challenging [18]. The problem is closely related to volume maximization in convex

*School of Mathematical and Statistical Sciences, Clemson University, Clemson, SC, USA. hhu2@clemson.edu
Research supported by the Air Force Office of Scientific Research under award number FA9550-23-1-0508.

†Department of Mathematics, Walla Walla University, 204 S College Ave, College Place, WA 99324, USA.

‡Department of Mathematics, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA. Research supported by National Science Foundation grant DMS 2348720.

§Department of Combinatorics and Optimization Faculty of Mathematics, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1; Research supported by The Natural Sciences and Engineering Research Council of Canada; www.math.uwaterloo.ca/~hwolkowi.

geometry—specifically, the volume of a simplex or parallelepiped spanned by a subset of vectors [7, 11, 12]. These geometric insights connect the MAXDET to low-rank approximation [10] and matrix sketching techniques.

In experimental design, particularly for linear and logistic regression, selecting a submatrix with maximum determinant corresponds to finding a D-optimal design—one that minimizes the volume of the confidence ellipsoid for the parameter estimates [25]. Applications include medical data modeling [24], constrained design construction [28], and categorical data analysis [30]. Classical and Bayesian formulations of optimal design problems have been well studied in the literature [27].

In machine learning and statistics, determinant-based selection criteria have been adopted for data summarization and diversity modeling using determinantal point processes [20]. Related problems also arise in active learning [31] and sparse modeling with budget constraints [26].

To cope with the combinatorial explosion of subset selection, numerous algorithms have been proposed. Greedy algorithms based on rank-revealing QR factorizations and convex relaxations using log-determinant functions have been effective [4, 5]. Matrix inequality constrained optimization frameworks, such as those in determinant maximization problems [29], have been used to model these tasks. Recent advances include primal-dual methods for optimization [9], submodular function maximization under matroid or partition constraints [22, 23], and proportional volume sampling [23].

In this paper, we explore various approaches for solving and approximating the MAXDET, where the size r coincides with the rank of the positive semidefinite matrix. We propose an efficient upper bound based on Hadamard’s inequality, which leads to a highly effective exact algorithm for solving small to moderate instances. This efficient upper bound uses an orthogonal projection method. In addition, we develop stronger upper bounds using semidefinite programming (SDP) relaxations. For larger problems we use a branch and bound technique which combined with the efficient upper bound seems to work quite effectively. We also apply the techniques to the NP-hard odd cycle packing problem.

Notation: We use $|\cdot|$ to denote both absolute value and cardinality, depending on the context. Define $\text{diag}(M): \mathcal{M}^n \rightarrow \mathbb{R}^n$ denotes the linear transformation on square matrices order n that yields the diagonal vector; the adjoint linear transformation is $\text{Diag}(v) = \text{diag}^*(v)$. We denote by e the vector of all ones of appropriate dimension.

Outline: The paper is organized as follows. We continue in Section 2 with the problem definition and relaxations based on conic representation of polynomial functions. Then in Section 3 we reformulate the problem using projections, which substantially strengthen the upper bounds. In Section 4 we present an efficient upper bound for the MAXDET problem based on the Hadamard inequality. This upper bound together with the projection technique yields an efficient branch-and-bound algorithm for solving the MAXDET problem. We also study the upper bound based on linear programming (LP), and its behavior under the projection technique, and we propose a tighter upper bound based on SDP. In Section 5, we present the numerical results of our branch-and-bound algorithm and the LP upper bounds.

2 Preliminaries

2.1 Problem definition

Let \mathbb{S}^n be the Euclidean space of real $n \times n$ symmetric matrices equipped with the *trace inner-product* $\langle A, B \rangle = \text{trace}(AB)$, and let \mathbb{S}_+^n denote the cone of positive semidefinite matrices. Fix

$$M \in \mathbb{S}_+^n, \text{ with } \text{rank}(M) = r < n. \quad (2.1)$$

77 For any subset $K \subseteq \{1, \dots, n\}$, we let M_K denote the principal submatrix of M consisting of
 78 elements from rows and columns indexed by K . We consider the following problem of maximizing
 79 the volume (or determinant).

80 **Problem 2.1** (MAXDET). *Given $M \in \mathbb{S}_+^n$, $\text{rank}(M) = r$, find a principal submatrix of M that*
 81 *maximizes the volume*

$$\begin{aligned} \max \quad & \det(M_K) \\ \text{s. t.} \quad & |K| = r \\ & K \subseteq \{1, \dots, n\}. \end{aligned} \tag{2.2}$$

82 It is often convenient to study an alternative formulation of Problem 2.1 using the factor-
 83 ization of M . Let $V \in \mathbb{R}^{n \times r}$ provide the full rank factorization $M = VV^T$. Let V_K denote the
 84 submatrix of V consisting of the rows indexed by K . Then we have $\det(M_K) = \det(V_K)^2$ and
 85 thus Problem 2.1 is equivalent to

$$\begin{aligned} \max \quad & |\det(V_K)| \\ \text{s. t.} \quad & |K| = r \\ & K \subseteq \{1, \dots, n\}. \end{aligned} \tag{2.3}$$

86 In this paper we assume that the matrix V in the factorization is given. We also note that the
 87 optimal value of (2.2) is the square of the optimal value of (2.3).

88 To compute an exact optimal solution, a branch-and-bound algorithm can be employed to
 89 recursively determine the subset of rows of V to be included in the final solution. We begin by
 90 formalizing the problem addressed at each node of the branch-and-bound algorithm. Specifically,
 91 we consider a generalized version of the original problems (2.2) and (2.3), where a subset of
 92 variables can be fixed to one, and others are fixed to zero. Rows of V corresponding to variables
 93 fixed at zero can be safely removed. For clarity, we focus on the case where a subset of variables
 94 is fixed to one. Given a subset $J \subseteq \{1, \dots, n\}$ with at most r elements, define the following
 95 variants of the problems (2.2) and (2.3).

- 96 1. The principal submatrix of $M = VV^T$ containing the rows and columns indexed by J that
 97 maximize the determinant is formulated as follows:

$$\begin{aligned} \delta(V, J) := \max \quad & \det(M_{J \cup K}) \\ \text{s. t.} \quad & K \subseteq \{1, \dots, n\}, \\ & J \cap K = \emptyset, \\ & |J \cup K| = r. \end{aligned} \tag{2.4}$$

- 98 2. The maximum absolute determinant of any $r \times r$ submatrix of V that contains the rows
 99 indexed by J . This can be formulated as follows:

$$\begin{aligned} \sqrt{\delta(V, J)} = \max \quad & |\det(V_{J \cup K})| \\ \text{s. t.} \quad & K \subseteq \{1, \dots, n\}, \\ & J \cap K = \emptyset, \\ & |J \cup K| = r. \end{aligned} \tag{2.5}$$

100 Note that $|J|$ can range from 0 to r , and we require $r - |J|$ additional rows to form a full-rank
 101 $r \times r$ submatrix.

2.2 Conic representations

In this section, we review some well-known results about conic representation of polynomial functions, see, e.g., [2, 14–16, 21]. Let $X \in \mathbb{S}_+^r$, and let $Z \in \mathbb{R}^{r \times r}$ be lower triangular. Then

$$\begin{pmatrix} X & Z \\ Z^T & \text{Diag}(\text{diag}(Z)) \end{pmatrix} \in \mathbb{S}_+^{2r} \implies \det(X) \geq \prod_{i=1}^r Z_{ii}. \quad (2.6)$$

In particular, there exists some lower triangular matrix Z satisfying (2.6) such that $\det(X) = \prod_{i=1}^r Z_{ii}$. Thus, if we need to find a matrix X with the maximum determinant, then we can alternatively maximize the product $\prod_{i=1}^r Z_{ii}$. A proof of this claim can be found on p. 150 of [3]. To linearize the product $\prod_{i=1}^r Z_{ii}$, we use the *exponential cone* which is defined as

$$K_{exp} := \left\{ (x_1, x_2, x_3) \in \mathbb{R}^3 \mid x_1 \geq x_2 e^{x_3/x_2}, x_2 > 0 \right\} \cup \left\{ (x_1, 0, x_3) \in \mathbb{R}^3 \mid x_1 \geq 0, x_3 \leq 0 \right\}.$$

We introduce a vector of variables $s \in \mathbb{R}^r$ and add the constraint

$$(Z_{ii}, 1, s_i) \in K_{exp} \text{ for } i = 1, \dots, r.$$

It follows from the definition of the exponential cone that $s_i \leq \log Z_{ii}$ for all $i = 1, \dots, r$. Therefore, we can model the maximum of $\prod_{i=1}^r Z_{ii}$ as the maximum of $\sum_{i=1}^r s_i$.¹

3 Reformulation via Projection

In this section, we describe a projection-based reformulation of the matrix V that is particularly useful within a branch-and-bound algorithm. This approach can significantly strengthen upper bounds in practice.

Without loss of generality, assume $J = \{1, \dots, k\}$ for some $k \leq r$. We define a new matrix $\tilde{V} \in \mathbb{R}^{n \times r}$, whose rows $\tilde{v}_1, \dots, \tilde{v}_n$ are computed via the following orthogonal projection procedure:

Algorithm 3.1 Orthogonal Projection Process

- 1: Set $\tilde{v}_1 \leftarrow v_1$.
 - 2: **for** $i = 2$ to k **do**
 - 3: Set \tilde{v}_i to the projection of v_i onto $\text{span}\{\tilde{v}_1, \dots, \tilde{v}_{i-1}\}^\perp$.
 - 4: **end for**
 - 5: **for** $i = k + 1$ to n **do**
 - 6: Set \tilde{v}_i to the projection of v_i onto $\text{span}\{\tilde{v}_1, \dots, \tilde{v}_k\}^\perp$.
 - 7: **end for**
-

Since the determinant is preserved under this projection process, we have

$$|\det(V_{J \cup K})| = |\det(\tilde{V}_{J \cup K})|$$

for all subsets $K \subseteq \{1, \dots, n\}$ such that $J \cap K = \emptyset$ and $|J \cup K| = r$. It follows that $\delta(V, J) = \delta(\tilde{V}, J)$. Although this is an equivalent reformulation in terms of determinant values, we will later show that various upper bounds on $\delta(\tilde{V}, J)$ are significantly tighter than the corresponding bounds on $\delta(V, J)$.

The projections in Algorithm 3.1 are computationally inexpensive and can be further optimized in the context of branch-and-bound.

¹We note that it is also possible to model $\prod_{i=1}^r Z_{ii}$ via second order cones.

1. In line 3, many projections can be reused to avoid redundant computation. For instance, suppose the current node in the branch-and-bound tree corresponds to $J = \{1, \dots, k\}$, and the matrix \tilde{V} has already been computed. Since the vectors $\tilde{v}_1, \dots, \tilde{v}_{k+1}$ are pairwise orthogonal, if the algorithm branches by setting $x_{k+1} = 1$, thereby updating $J \leftarrow \{1, \dots, k+1\}$ in the child node, then the projections of v_i onto $\text{span}\{\tilde{v}_1, \dots, \tilde{v}_{i-1}\}^\perp$ for $i = 2, \dots, k+1$ need not be recomputed, provided they are cached appropriately from the parent node.
2. In line 6, since $\tilde{v}_1, \dots, \tilde{v}_k$ are orthogonal, the projection of v_i onto the orthogonal complement of their span has the explicit formula:

$$\tilde{v}_i = v_i - \sum_{j=1}^k \frac{\tilde{v}_j^\top v_i}{\tilde{v}_j^\top \tilde{v}_j} \tilde{v}_j.$$

This expression is both computationally efficient and numerically stable due to the orthogonality of the basis vectors.

4 Upper bounds

We now consider several efficient techniques for obtaining upper bounds. This includes using the Hadamard inequality as well as linear programming and more general conic programming.

4.1 Upper bounds based on Hadamard-inequality

In this section, we present an efficient upper bound for the maximum determinant problem. Our approach leverages the Hadamard inequality [13], which states that the absolute value of the determinant of a square matrix is bounded above by the product of the norms of its rows. To tighten this bound, we also reformulate the rows of the matrix V via the projection technique as discussed in the last section. Numerical results demonstrate that the proposed Hadamard-based upper bound is highly effective, enabling the identification of optimal solutions within reasonable time when embedded in a branch-and-bound framework.

Let $V \in \mathbb{R}^{n \times r}$ and $J \subseteq \{1, \dots, n\}$ be given. For any subset $K \subseteq \{1, \dots, n\}$ such that

$$J \cap K = \emptyset, \quad |J \cup K| = r, \quad (4.1)$$

the submatrix $V_{J \cup K}$ is of size $r \times r$. Applying the Hadamard inequality yields the bound

$$|\det(V_{J \cup K})| \leq \prod_{i \in J \cup K} \|v_i\|,$$

where v_i denotes the i -th row of V . This leads to the following upper bound on $\delta(V, J)$:

$$\delta_H(V, J) := \prod_{i \in J} \|v_i\| \cdot \max \left\{ \prod_{i \in K} \|v_i\| \mid K \text{ satisfies (4.1)} \right\}. \quad (4.2)$$

Proposition 4.1. *Let $V \in \mathbb{R}^{n \times r}$ be given, and let $J \subseteq \{1, \dots, n\}$ with $|J| \leq r$. Let $\tilde{V} \in \mathbb{R}^{n \times r}$ be the matrix obtained by applying the orthogonal projection process described in Algorithm 3.1 to V with respect to J . Then,*

$$\delta_H(\tilde{V}, J) \leq \delta_H(V, J).$$

Proof. It is clear from the description of Algorithm 3.1 that $\|\tilde{v}_i\| \leq \|v_i\|$ for all i , the inequality follows. \square

The bound $\delta_H(\tilde{V}, J)$ is typically much tighter than $\delta_H(V, J)$. In practice, the projection significantly improves the quality of the bound.

4.2 An LP relaxation

For huge problem instances, we may not be able to find the optimal solution. In this case, it is important to measure the gap between the optimal value and the objective value of the best feasible solution we found. This gives the users a good idea about the quality of the feasible solution. To this end, we derive and analyze stronger upper bounds based on convex optimization techniques.

Let $V \in \mathbb{R}^{n \times r}$ and $J \subseteq \{1, \dots, n\}$ be given. Define the set

$$\Delta_n = \{x \in \mathbb{R}^n \mid e^T x = r, x \in [0, 1]^n\}. \quad (4.3)$$

A binary programming formulation for the original problem (2.4) is given as follows:

$$\begin{aligned} \max \quad & \det(V^T \text{Diag}(x)V) \\ \text{s. t.} \quad & x \in \Delta_n \cap \{0, 1\}^n, \\ & x_i = 1 \text{ for } i \in J. \end{aligned} \quad (4.4)$$

Indeed, if x^* is an optimal solution to (4.4), then $K := \{i \in \{1, \dots, n\} : x_i^* = 1\}$ is optimal for (2.4).

If we discard the binary constraints, then we obtain the well-known LP relaxation whose optimal value is denoted by $\delta_{LP}(V, J)$,

$$\begin{aligned} \max \quad & \det(V^T \text{Diag}(x)V) \\ \text{s. t.} \quad & x \in \Delta_n, \\ & x_i = 1 \text{ for } i \in J. \end{aligned} \quad (4.5)$$

It is fairly easy to see that $\delta_{LP}(V, J)$ is indeed an upper bound for the maximum of (4.4). Indeed, any binary feasible solution is also feasible for the LP relaxation. We note that $\log \det(V^T \text{Diag}(x)V)$ is a concave function and thus the maximization problem (4.5) is a convex optimization problem.

We prove that the projection technique also works for the LP relaxation (4.5), namely, if \tilde{V} is the matrix obtained from V by applying the projection procedure described in Algorithm (3.1), then $\delta_{LP}(\tilde{V}, J) \leq \delta_{LP}(V, J)$. To prove this, we derive the following useful linear algebra results.

Lemma 4.2.

$$\det \begin{pmatrix} \alpha + a & b^T \\ c & D \end{pmatrix} = \alpha \det D + \det \begin{pmatrix} a & b^T \\ c & D \end{pmatrix}.$$

Proof. Note that the first column can be written as $\alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} a \\ c \end{pmatrix}$. Applying the multilinearity of the determinant, we obtain the desired equation. \square

Proposition 4.3. Let $A \in \mathbb{R}^{k \times k}$ and

$$\begin{pmatrix} B & C \\ C^T & D \end{pmatrix} \in \mathbb{R}^{n \times n}$$

be positive semidefinite. Then

$$\det \begin{pmatrix} A + B & C \\ C^T & D \end{pmatrix} \geq \det A \det D.$$

171 *Proof.* We proceed by induction on k , the size of A . When $k = 1$, the result follows directly from
 172 Lemma 4.2. Suppose that the result has been proven for A up to size $k - 1$.

Without loss of generality, assume that A is a diagonal matrix, i.e., $A = \text{Diag}(a_i)_{i=1}^k$, where $a_i \geq 0$, $i = 1, \dots, k$. Let $P : \mathbb{C}^k \rightarrow \mathbb{C}^{k-1}$ be the projection on the last $k - 1$ entries. By the case $k = 1$, we have that

$$\det \begin{pmatrix} A+B & C \\ C^T & D \end{pmatrix} \geq a_1 \det \begin{pmatrix} PAP^T + PBP^T & CP^T \\ PC^T & D \end{pmatrix}.$$

By the induction assumption, we get that

$$\det \begin{pmatrix} PAP^T + PBP^T & CP^T \\ PC^T & D \end{pmatrix} \geq \det(PAP^T) \det(D).$$

173 Since, $\det A = a_1 \det(PAP^T)$, the result follows. \square

174 Now we are ready to prove the main theoretical result.

Theorem 4.4. *Let $V \in \mathbb{R}^{n \times r}$ and $J \subseteq \{1, \dots, n\}$ be given. Let \tilde{V} be the matrix obtained from V by applying the projection procedure in Algorithm 3.1. Then*

$$\delta_{LP}(\tilde{V}, J) \leq \delta_{LP}(V, J).$$

Proof. Without loss of generality, assume that the rows in V indexed by J are linearly independent and $J = \{1, \dots, k\}$ for some $k < r$. Note that the first k rows of \tilde{V} are orthogonal. We may multiply \tilde{V} on the right with a unitary matrix U , and make the first k rows of \tilde{V} to be of the form $\begin{pmatrix} \tilde{\Delta} & 0 \end{pmatrix}$, where $\tilde{\Delta}$ is a $k \times k$ positive definite diagonal matrix. Indeed, if w_1^T, \dots, w_k^T are the first k rows of V and are all nonzero, one would take

$$U = \begin{pmatrix} \frac{w_1}{\|w_1\|} & \cdots & \frac{w_k}{\|w_k\|} & u_{k+1} & \cdots & u_r \end{pmatrix},$$

175 where $\{u_{k+1}, \dots, u_r\}$ is an orthonormal basis for $(\text{span}\{w_1, \dots, w_k\})^\perp$.

Thus we have that we may assume that

$$\tilde{V} = \begin{pmatrix} \tilde{\Delta} & 0 \\ 0 & B \end{pmatrix},$$

for some $(n - k) \times (r - k)$ matrix B . The way, \tilde{V} was constructed from V , gives that

$$V = \begin{pmatrix} L\tilde{\Delta} & 0 \\ A & B \end{pmatrix},$$

176 where L is a lower triangular matrix with 1's on the diagonal, and A is of size $(n - k) \times k$.

Suppose that x^* yields the maximum $\delta_{LP}(\tilde{V}, J)$. Note that $x_j^* = 1$ for $j = 1, \dots, k$. Let $y = (x_i)_{i=k+1}^n$ be the vector consisting of the remaining $n - k$ entries of x^* . Thus $\delta_{LP}(\tilde{V}, J) = \det \tilde{\Delta}^2 \det(B^T \text{Diag}(y) B)$. Now

$$\begin{aligned} \delta_{LP}(V, J) &\geq \det(V^T \text{Diag}(x^*) V) \\ &= \det \begin{pmatrix} \tilde{\Delta} L^T L \tilde{\Delta} + A^T \text{Diag}(y) A & A^T \text{Diag}(y) B \\ B^T \text{Diag}(y) A & B^T \text{Diag}(y) B \end{pmatrix} \\ &\geq \det(\tilde{\Delta} L^T L \tilde{\Delta}) \det(B^T \text{Diag}(y) B) \\ &= \delta_{LP}(\tilde{V}, J) \end{aligned}$$

177 where the last inequality follows from Proposition 4.3 and we used that $\det(\tilde{\Delta} L^T L \tilde{\Delta}) = \det \tilde{\Delta}^2$
 178 due to $\det L = 1$. \square

179

4.3 SDP relaxations

In this section, we explore stronger convex relaxations for the MAXDET problem. We first provide an example to show that the standard lifting procedure for the binary feasible set (4.4) does not yield anything stronger than the LP relaxation (4.5). To construct a stronger relaxation, we present a conic programming formulation for the problem (4.4). The conic formulation is a direct application of two well-known conic representation results in the literature. This conic formulation is significantly more expensive to solve than the LP relaxation.

We first consider the standard procedure in constructing SDP relaxations for zero-one programming problems. Let $Y \in \mathbb{S}^{n+1}$ be a matrix variable indexed by $\{0, 1, \dots, n\}$. It is natural to attempt to strengthen the LP relaxation (4.5) by introducing a matrix variable Y and incorporating additional valid constraints based on the reformulation–lifting technique. For example, the following is an SDP relaxation

$$\begin{aligned}
\max \quad & \det(V^T \text{Diag}(x)V) \\
\text{s. t.} \quad & x \in \Delta_n \\
& x_i = 1 \text{ for } i \in J \\
& x = \text{diag}(X) \\
& Y \begin{pmatrix} -r \\ e \end{pmatrix} = 0 \\
& Y = \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \in \mathbb{S}_+^{n+1}.
\end{aligned} \tag{4.6}$$

Unfortunately this does not lead to a stronger upper bound than the LP relaxation, as we show in the next result.

Lemma 4.5. *The optimal values of (4.5) and (4.6) are the same.*

Proof. If (x, Y) is feasible for (4.6), then x is feasible for (4.5) with the same objective value. Conversely, assume x is feasible for (4.5). Since $x \in \Delta_n$ we can write x as a convex combination of the vertices of Δ_n . Note that the vertices of Δ_n are binary. Thus, $x = \sum_{i=1}^n \lambda_i x^i$ for some $x^i \in \Delta_n \cap \{0, 1\}^n$. For each x^i , the matrix

$$Y^i = \begin{pmatrix} 1 \\ x^i \end{pmatrix} \begin{pmatrix} 1 \\ x^i \end{pmatrix}^T$$

satisfies all the constraints in (4.6). Define $Y := \sum_{i=1}^n \lambda_i Y^i$. We have (x, Y) is feasible for (4.6), and it has the same objective. \square

The key reason why (4.6) does not provide a stronger relaxation is that the extreme points of Δ_n are exactly the binary solutions in $\Delta_n \cap \{0, 1\}^n$. As additional PSD constraints only strengthen the feasible set, it does not affect the objective function and thus it does not yield any improvements. To obtain stronger relaxations, we need to take the non-linear objective function into consideration. To this end, we first reformulate the non-linear objective function using linear conic constraints.

A lower triangular matrix of size $r \times r$ contains $\binom{r}{2}$ entries. We define the linear operator $\mathcal{L} : \mathbb{R}^{\binom{r}{2}} \rightarrow \mathbb{R}^{r \times r}$ that maps any given vector $z \in \mathbb{R}^{\binom{r}{2}}$ into a lower triangular matrix. We also assume that the first r entries in z corresponds to the r diagonal entries in $\mathcal{L}(z)$. Denote by $\text{Diag}(z_1, \dots, z_r)$ the diagonal matrix whose diagonal entries are z_1, \dots, z_r . Define the linear operator $\mathcal{V} : \mathbb{R}^{n+\binom{r}{2}} \rightarrow \mathbb{S}^{2r}$ as follows:

$$\mathcal{V}(x, z) := \begin{pmatrix} V^T \text{Diag}(x)V & \mathcal{L}(z) \\ \mathcal{L}(z)^T & \text{Diag}(z_1, \dots, z_r) \end{pmatrix}.$$

203 Following the discussion in Section 2.2, the binary formulation (4.4) can be equivalently formu-
 204 lated as follow:

$$\begin{aligned}
 & \max \quad e^T s \\
 & \text{s. t.} \quad \mathcal{V}(x, z) \in \mathbb{S}_+^{2r} \\
 & \quad (z_i, 1, s_i) \in K_{exp} \text{ for } i = 1, \dots, r \\
 & \quad z \in \mathbb{R}^{\binom{r}{2}}, s \in \mathbb{R}^r, x \in \Delta_n \cap \{0, 1\}^n.
 \end{aligned} \tag{4.7}$$

205 Indeed, we have $\prod_{i=1}^r z_i = \det(V^T \text{Diag}(x)V)$ and $s_i \leq \log z_i$. Thus, at the optimum we have

$$e^T s = \sum \log z_i = \log \det V^T \text{Diag}(x)V.$$

206 If we relax the binary constraint, then we obtain an equivalent formulation for the LP re-
 207 laxation (4.5). We propose a stronger relaxation as follows. Let $Y \in \mathbb{S}_+^{n+1}$ be a matrix variable
 208 whose rows and columns are indexed by $\{0, 1, \dots, n\}$. Denote by Y_{ij} the entry of Y indexed by
 209 the i -th row and j -th column. Denote by y^i the i -th column of Y without the 0-th entry. Thus
 210 y^i is an n -dimensional column vector. Let z^0, \dots, z^n be vectors of variables in $\mathbb{R}^{\binom{r}{2}}$. Let $s \in \mathbb{R}^r$
 211 be a vector of variables with r entries s_1, \dots, s_r .

$$\begin{aligned}
 & \max \quad e^T s \\
 & \text{s. t.} \quad \mathcal{V}(y^i, z^i) \in \mathbb{S}_+^{2r} \text{ for } i = 0, \dots, n \\
 & \quad \mathcal{V}(y^0 - y^i, z^0 - z^i) \in \mathbb{S}_+^{2r} \text{ for } i = 1, \dots, n \\
 & \quad (z_j^0, 1, s_j) \in K_{exp} \text{ for } j = 1, \dots, r \\
 & \quad Y \begin{pmatrix} -r \\ e \end{pmatrix} = 0 \\
 & \quad Y_{0i} - Y_{ii} = 0 \text{ for } i = 1, \dots, n \\
 & \quad Y \geq 0 \\
 & \quad z^i \in \mathbb{R}^{\binom{r}{2}} \text{ for } i = 1, \dots, n \\
 & \quad s \in \mathbb{R}^r, y^0 \in \Delta_n \\
 & \quad Y \in \mathbb{S}_+^{n+1}.
 \end{aligned} \tag{4.8}$$

212 We can easily verify that (4.8) is a relaxation for (4.7). Let (x, z, s) be any feasible solution for
 213 the binary formulation (4.7). Without loss of generality, assume that $x_1 = \dots = x_r = 1$ and
 214 $x_{r+1} = \dots = x_n = 0$. Let $y^0 = x$ and $z^0 = z$. Let $y^i = x$ and $z^i = z$ for $i = 1, \dots, r$, and $y^i = 0$
 215 and $z^i = 0$ for $i = r + 1, \dots, n$. Then y^i, z^i, s form a feasible solution for (4.8) with the same
 216 objective value. Indeed, we have $\mathcal{V}(y^i, z^i)$ and $\mathcal{V}(y^0 - y^i, z^0 - z^i)$ are either a matrix of zeros or
 217 the same matrix as $\mathcal{V}(y^0, z^0)$. Thus, they are positive semidefinite. The remaining constraints
 218 can be verified easily as well.

219 Although (4.8) contains both exponential cone and positive semidefinite cone constraints,
 220 we abuse the notation and still call it an SDP relaxation. While the SDP relaxation (4.8) is
 221 strictly stronger than the LP relaxation (4.7), it is also much more expensive. For example, (4.8)
 222 involves a matrix variable of order $n + 1$, while (4.7) only involves a matrix variable of order r .
 223 For problems in practice, n is usually a huge number, and thus, it is very impractical to solve
 224 (4.8). We consider the SDP relaxation (4.8) as a theoretical contribution, and it has limited
 225 computational advantage.

226 5 Numerics

227 In this section, we present our numerical experiments. All computations were performed on a
 228 Mac Studio (2023) equipped with an Apple M2 Ultra chip, 128 GB of RAM, and running macOS

14.1.1 (build 23B81). The branch-and-bound algorithm was implemented in MATLAB R2024b. The LP relaxation (4.5) was formulated as a linear SDP and solved using MOSEK version 11 [1].

We report the objective value as $\log \det(V^\top \text{Diag}(x)V)$ or $\log_2 \det(V^\top \text{Diag}(x)V)$, as the determinant can take extremely large values in high-dimensional settings.

We solve the problem using the branch-and-bound algorithm, which incorporates Hadamard-type inequality bounds described in Section 4.1. The best objective value obtained from the branch-and-bound process is recorded as the **lower bound (LB)**. Additionally, we solve the LP relaxation (4.5) to obtain an **upper bound (UB)** on the optimal value.

To evaluate the quality of the solutions, we compute the normalized duality gap:

$$\text{GAP} = \frac{|\text{UB} - \text{LB}|}{\max\{|\text{UB}|, |\text{LB}|, \varepsilon\}},$$

where $\varepsilon = 10^{-8}$ is a small constant used to prevent division by zero.

A time limit of 10 minutes is imposed for the branch-and-bound algorithm. If the algorithm certifies optimality within this time, the corresponding LB is marked with an asterisk (*). For both the branch-and-bound and LP relaxation approaches, we report the runtime in seconds.

5.1 The UCI Machine Learning Repository

We evaluate our algorithm using datasets from the UCI Machine Learning Repository [8]. Each dataset is represented by a matrix $V \in \mathbb{R}^{n \times r}$, where each row corresponds to an observation and each column to a feature. We restrict our experiments to datasets that satisfy the following two conditions: (1) the matrix V contains only numerical values, and (2) the number of observations exceeds the number of features, i.e., $n > r$. If V contains linearly dependent columns, we extract a subset of linearly independent columns using QR factorization. With slight abuse of notation, we continue to denote the resulting matrix by V .

Tables 5.1 and 5.2 summarize the datasets along with problem dimensions (n, r) , objective values obtained by both methods (LB from branch-and-bound, UB from LP relaxation), computation times in seconds, and the normalized duality gap.

Observations. Based on the numerical results, we highlight the following observations:

- The branch-and-bound algorithm successfully solves several small- and medium-sized instances to optimality within seconds (e.g., `Lenses`, `Fertility`, `Iris`, `Hayes-Roth`), as indicated by the asterisk next to the LB.
- For larger or more complex datasets (e.g., `Parkinsons`, `Image_Segmentation`, `SPECTF_Heart`), the branch-and-bound method often reaches the time limit without proving optimality, although it still provides meaningful lower bounds.
- The LP relaxation is highly efficient in most cases, typically completing in under one second. However, its quality varies significantly across datasets. In two large-scale instances, the LP solver exceeded our machine’s memory or time constraints; these are labeled as N.A.
- The duality gap is negligible when the branch-and-bound algorithm reaches optimality or a tight bound, indicating the relaxation is informative in those cases. Conversely, large duality gaps (e.g., `Vertebral_Column`, `User_Knowledge`) suggest weaker LP relaxations and emphasize the need for exact methods.

Overall, the results demonstrate the effectiveness of the proposed approach on small to moderate-sized datasets and illustrate the trade-off between computational time and solution quality for larger problems.

Dataset	n	r	Branch-and-Bound		LP relaxation		GAP
			LB	Time	UB	Time	
Challenger_USA_Space_Shuttle_O-Ring	23	4	-17.3036(*)	0.72	-17.1441	0.01	0.01
Lenses	24	3	-0.94001(*)	0.41	-0.94001	0.01	0.00
Soybean_(Small)	47	20	-48.5653	600	-44.3561	0.12	0.09
Daily_Demand_Forecasting_Orders	60	12	-136.6481	600	-125.9895	0.07	0.08
Cervical_Cancer_Behavior_Risk	72	19	-11.748	600	-8.1748	0.12	0.30
Hepatitis	80	19	-168.6379	600	-165.6924	0.51	0.02
Fertility	100	9	7.4072(*)	76.72	8.3906	0.02	0.12
Zoo	101	16	-52.8589	600	-50.5052	0.05	0.04
Breast_Cancer_Coimbra	116	9	-57.447(*)	11.19	-57.1875	0.05	0.00
Primary_Tumor	132	17	-16.0921	600	-12.1534	0.07	0.24
Higher_Education_Students_Performance_Evaluation	145	31	-56.8203	600	-46.5442	0.24	0.18
Iris	150	4	-9.6767(*)	1.09	-9.3734	0.02	0.03
Hayes-Roth	160	4	-0.29403(*)	0.53	-0.1878	0.02	0.36
Wine	178	13	-147.6318	600	-145.8124	0.43	0.01
Breast_Cancer_Wisconsin_(Prognostic)	194	33	-556.5208	600	-438.696	1.11	0.21
Parkinsons	195	20	-332.9395	600	-254.6942	0.41	0.24
Image_Segmentation	210	19	-280.9973	600	-190.3849	0.29	0.32
Glass_Identification	214	9	-61.7463	600	-61.0086	0.21	0.01
Soybean_(Large)	266	35	-70.8578	600	-61.1069	0.37	0.14
SPECTF_Heart	267	44	-61.7682	600	-51.4834	1.47	0.17
SPECT_Heart	267	22	28.2914	600	34.7918	0.11	0.19
Statlog_(Heart)	270	13	-105.021	600	-102.7533	0.27	0.02
Heart_Disease	297	13	-104.9455	600	-102.7195	0.22	0.02
Heart_Failure_Clinical_Records	299	12	-236.0395	600	-179.6787	0.23	0.24
Haberman's_Survival	306	3	-2.2252(*)	0.02	-2.2252	0.03	0.00
Vertebral_Column	310	6	-54.6542	600	-27.9576	0.12	0.49
Ecoli	336	7	-4.7142(*)	1.07	-4.221	0.06	0.10
Land_Mines	338	3	0.35537(*)	0.16	0.61624	0.03	0.42
Liver_Disorders	345	5	-9.6506(*)	0.29	-9.58	0.05	0.01
Ionosphere	351	33	80.2523	600	87.3991	0.87	0.08
Dermatology	358	34	-205.4	600	-195.2633	0.48	0.05
Auto_MPG	392	7	-66.0296	600	-65.7233	0.24	0.00
User_Knowledge_Modeling	403	5	-0.56247(*)	0.21	-0.20268	0.05	0.64
Real_Estate_Valuation	414	6	-70.8655	600	-58.7011	0.20	0.17
MONK's_Problems	432	6	-4.1125(*)	52.93	-3.462	0.07	0.16
Wholesale_customers	440	7	-30.8299	600	-26.1566	0.25	0.15
Breast_Cancer_Wisconsin_(Diagnostic)	569	30	-516.6465	600	-420.3534	2.43	0.19
Balance_Scale	625	4	1.1087(*)	0.16	1.2756	0.10	0.13
Breast_Cancer_Wisconsin_(Original)	683	9	2.0562(*)	29.06	2.8805	0.24	0.29
Statlog_(Australian_Credit_Approval)	690	14	-236.1593	600	-207.0687	0.86	0.12
National_Poll_on_Healthy_Aging_(NPHA)	714	14	-19.859	600	-17.265	0.39	0.13
Absenteeism_at_work	740	19	-124.532	600	-118.4592	1.35	0.05
Blood_Transfusion_Service_Center	748	3	-20.546(*)	0.18	-20.5458	0.30	0.00
Energy_Efficiency	768	7	-64.8916	600	-64.2346	0.72	0.01
Mammographic_Mass	830	5	-20.8851(*)	2.41	-20.632	0.37	0.01
Statlog_(Vehicle_Silhouettes)	845	18	-126.1669	600	-122.8075	3.13	0.03

Table 5.1: Summary of UCI datasets and results.

Dataset	n	r	Branch-and-Bound		LP relaxation		GAP
			LB	Time	UB	Time	
Raisin	900	7	-101.3097	600	-80.2776	0.92	0.21
Maternal_Health_Risk	1014	6	-13.7333(*)	510.77	-13.6184	0.46	0.01
Concrete_Compressive_Strength	1030	8	-24.6657	600	-22.9041	0.59	0.07
Diabetic_Retinopathy_Debrecen	1151	18	-139.2206	600	-135.1317	4.17	0.03
Website_Phishing	1353	9	17.5505	600	18.3885	0.94	0.05
Banknote_Authentication	1372	4	-3.9593(*)	2.48	-3.8468	0.56	0.03
Hepatitis_C_Virus_(HCV)_for_Egyptian_patients	1385	28	-531.9193	600	-358.6365	5.68	0.33
Contraceptive_Method_Choice	1473	9	-42.9772	600	-41.4642	1.57	0.04
Yeast	1484	8	-4.8033(*)	156.85	-4.1933	0.69	0.13
Airfoil_Self_Noise	1503	5	-73.1905	600	-57.2507	1.49	0.22
Drug_Consumption_(Quantified)	1885	12	0.18583	600	2.089	2.33	0.91
Steel_Plates_Faults	1941	27	-695.4052	600	-344.2675	7.07	0.50
Auction_Verification	2043	7	-40.9852	600	-40.1057	2.15	0.02
Cardiotocography	2126	20	-173.6264	600	-147.7546	8.46	0.15
AIDS_Clinical_Trials_Group_Study_175	2139	23	-269.8734	600	-258.93	10.31	0.04
National_Health_and_Nutrition_Health_Survey_2013-2014_	2278	7	-43.3426	600	-42.9178	4.22	0.01
Statlog_(Image_Segmentation)	2310	19	-270.7187	600	-184.6101	9.37	0.32
Iranian_Churn	3150	13	-161.5337	600	-153.2374	13.87	0.05
Ozone_Level_Detection	3695	72	-1021.1334	601	-963.1435	86.59	0.06
Rice_(Cammeo_and_Osmancik)	3810	7	-90.6916	600	-78.7753	15.77	0.13
Predict_Students'_Dropout_and_Academic_Success	4424	36	-459.3387	600	-437.3949	25.70	0.05
Spambase	4601	57	-809.8656	600	-785.3801	32.93	0.03
Waveform_Database_Generator_(Version_1)	5000	21	-14.459	600	-7.214	6.27	0.50
Page_Blocks_Classification	5473	10	-105.606	600	-92.8885	19.13	0.12
Optical_Recognition_of_Handwritten_Digits	5620	62	-5.1735	601	23.6513	45.87	1.22
Parkinsons_Telemonitoring	5875	19	-284.7582	600	-234.9981	22.02	0.17
Statlog_(Landsat_Satellite)	6435	36	-106.3612	600	-90.3697	39.49	0.15
Wine_Quality	6497	11	-95.0783	600	-94.1656	43.71	0.01
Musk_(Version_2)	6598	166	-332.7221	601	-245.6389	814.92	0.26
Taiwanese_Bankruptcy_Prediction	6819	24	-6.52	601	-1.6099	25.65	0.75
ISOLET	7797	617	1296.2834	609	N.A.	N.A.	N.A.
Combined_Cycle_Power_Plant	9568	4	-20.0107	600	-19.0065	29.76	0.05
Electrical_Grid_Stability_Simulated_Data_	10000	11	-23.6946	600	-21.217	31.47	0.10
Pen-Based_Recognition_of_Handwritten_Digits	10992	16	3.4228	600	8.5114	38.97	0.60
Phishing_Websites	11055	30	82.465	600	93.1696	138.58	0.11
Dry_Bean	13611	16	-391.885	601	-240.8034	153.03	0.39
EEG_Eye_State	14980	14	-183.4523	600	-158.72	181.37	0.13
HTRU2	17898	8	-47.5217	601	-46.257	214.40	0.03
MAGIC_Gamma_Telescope	19020	10	-47.9973	600	-45.7039	363.05	0.05
Polish_Companies_Bankruptcy	19967	65	-1361.4574	604	-922.5883	795.90	0.32
Letter_Recognition	20000	16	-8.9559	600	-5.1585	184.55	0.42
Superconductivity_Data	21263	81	-1037.7763	604	-905.8072	3503.56	0.13
NATICUSdroid_(Android_Permissions)	29332	85	128.0883	600	165.7042	1547.99	0.23
Default_of_Credit_Card_Clients	30000	23	-270.6213	600	-186.0217	1054.13	0.31
Online_News_Popularity	39644	55	-1126.6942	600	N.A.	N.A.	N.A.

Table 5.2: Summary of UCI datasets and results.

5.2 Odd Cycle Packing (OCP)

We also apply our algorithm to the odd cycle packing problem as this problem was used to show that MAXDET is NP-hard; see [7]. Given a simple undirected graph, we want to find a maximum family of vertex-disjoint odd cycles. For a given graph G with node-edge incidence matrix A_G , then, for every odd cycle C of G , the square submatrix of A_G with rows corresponding to the nodes of C and columns corresponding to the edges of G has determinant ± 2 . See e.g., [6].

We choose a random simple undirected graph G with incidence matrix $V^T = A_G$. We then apply our relaxed problem of maximizing the determinant.

We use the following MATLAB code to generate the random graphs.

```

G = graph(true(r), 'omitselfloops');
p = randperm(numedges(G), n);
G = graph(G.Edges(p, :));

```

```
V = full(abs(incidence(G)))';
```

n	r	Branch-and-Bound		LP relaxation		GAP
		LB	Time	UB	Time	
10	5	2(*)	0.01	4.3399	0.00	0.54
11	6	4(*)	0.00	4.7549	0.00	0.16
12	6	4(*)	0.00	4.937	0.00	0.19
13	7	4(*)	0.00	5.4449	0.00	0.27
14	7	4(*)	0.00	5.5565	0.00	0.28
15	8	4(*)	0.01	5.7443	0.00	0.30
16	8	4(*)	0.01	6.2	0.00	0.35
17	9	4(*)	0.03	6.5274	0.01	0.39
18	9	6(*)	0.03	7.3216	0.00	0.18
19	10	2(*)	0.40	6.2492	0.01	0.68
20	10	4(*)	0.13	7.3204	0.01	0.45
21	11	4(*)	0.23	7.5539	0.01	0.47
22	11	6(*)	0.07	8.4524	0.01	0.29
23	12	4(*)	0.51	7.6259	0.01	0.48
24	12	6(*)	0.29	9.008	0.01	0.33
25	13	6(*)	0.55	8.4247	0.02	0.29
26	13	4(*)	1.99	8.2512	0.02	0.52
27	14	6(*)	1.29	9.4815	0.01	0.37
28	14	6(*)	2.14	9.7649	0.01	0.39
29	15	6(*)	7.71	9.7195	0.01	0.38
30	15	8(*)	8.28	11.4575	0.01	0.30
31	16	8(*)	3.21	11.5478	0.02	0.31
32	16	6(*)	26.26	11.4689	0.01	0.48
33	17	8(*)	27.10	12.0676	0.02	0.34
34	16	10(*)	2.49	12.6579	0.01	0.21
35	18	6(*)	206.69	11.5627	0.07	0.48
36	18	8(*)	69.62	13.2113	0.02	0.39
37	19	6(*)	150.38	11.6104	0.05	0.48
38	19	6(*)	336.25	12.1435	0.07	0.51
39	20	8(*)	440.52	13.1572	0.05	0.39
40	20	8	600	14.7234	0.02	0.46
41	21	10(*)	243.98	14.8876	0.02	0.33
42	21	8	600	13.6182	0.06	0.41
43	22	8	600	15.1925	0.02	0.47
44	22	6	600	13.4259	0.12	0.55
45	23	10	600	14.8573	0.04	0.33
46	23	6	600	15.7472	0.03	0.62
47	24	8	600	15.2133	0.13	0.47
48	24	8	600	16.2876	0.09	0.51
49	25	8	600	16.1379	0.06	0.50
50	25	10	600	17.3448	0.10	0.42

Table 5.3: Summary of odd cycle datasets and results in log 2 base.

283 6 Statements and Declarations

284 **Competing Interests:** This work was supported in part by the Air Force Office of Scientific
285 Research under award number FA9550-23-1-0508 (Hao Hu), by the National Science Foundation
286 under grant DMS 2348720 (Hugo J. Woerdeman), and by the Natural Sciences and Engineering
287 Research Council of Canada (Henry Wolkowicz). The authors have no relevant financial or
288 non-financial interests to disclose.

289 **Data availability:** The datasets analysed during the current study were generated randomly,
290 and the procedure for their generation is fully described in the article. We also use previously
291 published datasets that are publicly available and fully cited within the article.

References

- [1] Mosek ApS. Mosek optimization toolbox for MATLAB. *User's Guide and Reference Manual, version, 4*, 2019. 10
- [2] A. Ben-Tal and A.S. Nemirovski. *Lectures on modern convex optimization*. MPS/SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001. Analysis, algorithms, and engineering applications. 4
- [3] Aharon Ben-Tal and Arkadi Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. SIAM, 2001. 4
- [4] Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009. 2
- [5] Giuseppe Della Riccia and Alexander Shapiro. Minimum rank and minimum trace of covariance matrices. *Psychometrika*, 47(3):443–448, 1982. 2
- [6] M. Di Summa, F. Eisenbrand, Y. Faenza, and C. Moldenhauer. On largest volume simplices and sub-determinants. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 315–323. SIAM, Philadelphia, PA, 2015. 12
- [7] Marco Di Summa, Friedrich Eisenbrand, Yuri Faenza, and Carsten Moldenhauer. On largest volume simplices and sub-determinants. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 315–323. SIAM, 2015. 1, 2, 12
- [8] Andrew Frank. UCI machine learning repository. <http://archive.ics.uci.edu/ml>, 2010. 10
- [9] M. Gonzalez-Lima, H. Wei, and H. Wolkowicz. A stable primal-dual approach for linear programming under nondegeneracy assumptions. *Computational Optimization and Applications*, 44(2):213–247, 2009. 2
- [10] S. Goreinov, I. Oseledets, D. Savostyanov, E. Tyrtyshnikov, and N. Zamarashkin. How to find a good submatrix. In *Matrix Methods: Theory, Algorithms and Applications*, pages 247–256. World Scientific, 2010. 2
- [11] S. Goreinov and E. Tyrtyshnikov. The maximal-volume concept in approximation by low-rank matrices. In *Structured matrices in mathematics, computer science, and engineering, I*, volume 280 of *Contemporary Mathematics*, pages 47–51. American Mathematical Society, 2001. 2
- [12] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudo-skeleton approximations. *Linear Algebra and its Applications*, 261(1-3):1–21, 1997. 2
- [13] B. Grone, C.R. Johnson, E. Marques de Sa, and H. Wolkowicz. Improving Hadamard's inequality. *Linear and Multilinear Algebra*, 16(1-4):305–322, 1984. 5
- [14] J.W. Helton and J. Nie. Sufficient and necessary conditions for semidefinite representability of convex hulls and sets. *SIAM J. Optim.*, 20(2):759–791, 2009. 4
- [15] J.W. Helton and J. Nie. Semidefinite representation of convex sets. *Math. Program.*, 122(1, Ser. A):21–64, 2010. 4

- [16] J.W. Helton and J. Nie. Semidefinite representation of convex sets and convex hulls. In *Handbook on semidefinite, conic and polynomial optimization*, volume 166 of *Internat. Ser. Oper. Res. Management Sci.*, pages 77–112. Springer, New York, 2012. 4
- [17] W.L. Jung, D. Torregrosa-Belen, and H. Wolkowicz. The ω -condition number: applications to preconditioning and low rank generalized Jacobian updating. *Comput. Optim. Appl.*, 91(1):235–282, 2025. 1
- [18] Leonid Khachiyan. On the complexity of approximating extremal determinants in matrices. *Journal of Complexity*, 11(1):138–153, 1995. 1
- [19] Chiu Ko, Ching Wah Lee, and Richard E. Stearns. The computational complexity of some problems in matrix theory. *SIAM Journal on Algebraic Discrete Methods*, 4(3):257–265, 1983. 1
- [20] Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012. 2
- [21] Arkadi Nemirovski. Advances in convex optimization: conic programming. In *International Congress of Mathematicians*, volume 1, pages 413–444, 2007. 4
- [22] Aleksandar Nikolov and Mohit Singh. Maximizing determinants under partition constraints. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 192–201. ACM, 2016. 2
- [23] Aleksandar Nikolov, Mohit Singh, and Kunal Talwar. Proportional volume sampling and approximation algorithms for a-optimal design. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, pages 671–680, 2015. 2
- [24] Liwen Ouyang, Daniel W. Apley, and Sanjay Mehrotra. A design of experiments approach to validation sampling for logistic regression modeling with error-prone medical records. *Journal of the American Medical Informatics Association*, 23(e1):e71–e78, 2016. 2
- [25] Friedrich Pukelsheim. *Optimal Design of Experiments*. SIAM, 2006. 2
- [26] Sathya Narayanan Ravi, Vamsi Ithapu, Sterling Johnson, and Vikas Singh. Experimental design on a budget for sparse linear models and applications. In *International Conference on Machine Learning*, pages 583–592. PMLR, 2016. 2
- [27] Paola Sebastiani and Henry P. Wynn. Maximum entropy sampling and optimal bayesian experimental design. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(1):145–157, 2000. 2
- [28] Dariusz Uciński. An algorithm for construction of constrained D-optimum designs. In *Stochastic Models, Statistics and Their Applications*, pages 461–468. Springer, 2015. 2
- [29] Lieven Vandenbergh, Stephen Boyd, and Shao-Po Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533, 1998. 2
- [30] Jie Yang, Liping Tong, and Abhyuday Mandal. D-optimal designs with ordered categorical data. *Statistica Sinica*, 27(4):1879–1902, 2017. 2
- [31] Wei Zheng, Ting Tian, and Xueqin Wang. Batch mode active learning for efficient parameter estimation. *arXiv preprint arXiv:2304.02741*, 2023. 2