

SDP bounds on the stability number via ADMM and intermediate levels of the Lasserre hierarchy

Lennart Sinjorgo^{*†} 

Renata Sotirov^{*} 

Juan C. Vera^{*} 

Abstract

We consider the Lasserre hierarchy for computing bounds on the stability number of graphs. The semidefinite programs (SDPs) arising from this hierarchy involve large matrix variables and many linear constraints, which makes them difficult to solve using interior-point methods. We propose solving these SDPs using the alternating direction method of multipliers (ADMM). When the second level of the Lasserre hierarchy for a given graph is intractable for the ADMM, we consider an intermediate-level relaxation of the hierarchy. To warm-start the ADMM, we use an optimal solution from the first level of the Lasserre hierarchy, which is equivalent to the well-known Lovász theta function. Additionally, we use this solution to determine which degree two monomials to add in the Lasserre hierarchy relaxation to obtain an intermediate level between 1 and 2. Computational results demonstrate that our approach yields strong bounds on the stability number, which are computable within reasonable running times. We provide the best-known bounds on the stability number of various graphs from the literature.

Keywords. semidefinite programming, stability number, ADMM, Lasserre hierarchy

1 Introduction

A stable set of a graph is a subset of vertices that are pairwise non-adjacent. Given an undirected graph G , the stable set problem is to determine a stable set of G of maximum cardinality. The stability number of G , denoted by $\alpha(G)$, is defined as the cardinality of a maximum stable set in G . Computing $\alpha(G)$ is NP-hard. Hence, unless $P = NP$, there is no polynomial time algorithm that computes it.

There exist different approaches for computing upper bounds on the stability number of a graph, and one of those is using semidefinite programming. The first SDP relaxation of the stable set problem is due to Lovász [26], who introduced the Lovász theta function of a graph G , denoted by $\vartheta(G)$. For any graph G , $\vartheta(G) \geq \alpha(G)$ and $\vartheta(G)$ can be computed in polynomial time up to fixed precision. Semidefinite programs (SDPs) that define the Lovász theta function can be strengthened by cutting planes, see e.g., [10, 16, 34, 37]. The paper by Pucher and Rendl [34] currently provides one of the strongest SDP-based bounds for the stable set problem.

Several hierarchical approaches can also be applied to construct relaxations of the stable set problem. Higher levels in the hierarchy correspond to stronger relaxations, which are also more difficult to solve due to the increased number of variables and constraints. Among the hierarchies that can be applied to the stable set problem are the Sherali-Adams hierarchy [38] (based on linear programming), the SDP-based hierarchy of Lovász-Schrijver [27], the Lasserre hierarchy [21], and the exact subgraph hierarchy (ESH) [1]. Much research has recently been devoted to the ESH for the stable set problem [12, 13, 14]. The computational results in those papers show that the ESH provides strong bounds within a reasonable computational time. It is known that the Lasserre hierarchy is stronger than the other hierarchies [23, 39]. Despite this, little research has been devoted to the practical performance of the Lasserre hierarchy for the stable set problem. A practical drawback of the Lasserre hierarchy is the order of the associated positive semidefinite (PSD) matrix variable: level k of the hierarchy involves a PSD matrix variable of order $\mathcal{O}(n^k)$, where n is the number of vertices in the graph.

The classical method for solving semidefinite programs (SDPs) is the interior-point method (IPM) [2, 31]. However, IPMs typically require large amounts of memory, which limits their applicability to the Lasserre hierarchy. The interior-point method is a second-order method that requires the construction and Cholesky factorization of an $m \times m$ Schur complement matrix, where m is the number of linear

^{*}CentER, Department of Econometrics and OR, Tilburg University, The Netherlands
{l.m.sinjorgo,r.sotirov,j.c.veralizcano}@tilburguniversity.edu

[†]corresponding author

equality constraints in the SDP relaxation. Constructing and storing this dense matrix requires substantial memory, particularly when m is large, such as in SDPs derived from the Lasserre hierarchy. The worst-case complexity of computing the Schur complement matrix is $\mathcal{O}(mp^3 + m^2p^2)$ [29], where p is the order of the PSD variable. Computational complexity may be reduced in cases where the constraint matrices exhibit special structures such as low rank, see e.g., [9, 18]. The Cholesky factorization for the Schur complement matrix requires $\mathcal{O}(m^3)$ operations, which becomes impractical for large values of m . These limitations of IPMs motivate the use of alternative methods that require less memory and bypass the Cholesky factorization.

The alternating direction method of multipliers (ADMM), see e.g., [5], is a first-order method that can be used to solve SDPs, and requires significantly less memory than IPMs. Each iteration of the ADMM algorithm for solving SDPs consists of three steps: the orthogonal projection onto the cone of positive semidefinite matrices, an orthogonal projection onto a polyhedral set, and a dual update step. The most memory intensive step of the ADMM is computing the eigenvalue decomposition of a symmetric PSD matrix of order p , in each main loop of the algorithm. The SDP relaxations of the stable set problem arising from the Lasserre hierarchy satisfy $m \gg p$, and the computational complexity of eigenvalue decomposition for a symmetric matrix of order p is $\mathcal{O}(p^3)$. Considering this, along with the fact that projections onto the polyhedral sets from the Lasserre relaxations can be performed efficiently, ADMMs appear more suitable than IPMs for computing Lasserre bounds for the stable set problem.

In this paper, we bridge the gap between theory and practice by using the ADMM to effectively compute Lasserre hierarchy bounds for the stability number. In particular, we compute bounds from intermediate levels of the Lasserre hierarchy for k between 1 and 2, including $k = 2$, on graphs with at most 300 vertices. However, we are not the first to consider intermediate levels of the Lasserre hierarchy; these have been employed in several other papers, see e.g., [3, 7, 8, 40, 43]. For the majority of graphs, we restrict ourselves to intermediate levels of the hierarchy due to practical limitations on the order of the considered PSD variable, which we cap at 2500. Eigenvalue decomposition for a symmetric matrix of that order can still be performed reasonably well, especially when single precision is used. The number of inequality constraints in the resulting SDP relaxation depends on the considered graph and is at most 2,510,148 in our experiments. Storing the $m \times m$ Schur complement matrix, with $m = 2,510,148$, requires 46,944.9 GB of RAM (!). Therefore, IPMs are intractable for solving the corresponding relaxations. Constructing an intermediate-level SDP relaxation of the Lasserre hierarchy for k between 1 and 2 requires selecting specific degree two monomials. These monomials of degree two, along with all monomials of degree one and the monomial of degree zero, form a basis used to derive the SDP relaxation. We present a basis selection method that exploits an SDP relaxation of the Lovász theta function. It is known that $\vartheta(G)$ corresponds to the first level of the Lasserre hierarchy applied to the stable set problem, see e.g., [23, Sect. 6]. Our ADMM algorithm incorporates a warm-starting approach to further improve performance. The computational results show that the upper bounds on $\alpha(G)$ computed here are competitive with the best SDP-based bounds for the stable set problem. Moreover, these bounds can be obtained using the ADMM within reasonable running times, specifically within one hour.

This paper is organized as follows. The notation is introduced in Section 1.1. We provide details of the Lasserre hierarchy for the stable set problem in Section 2. In Section 3, we show how to apply the ADMM to the SDPs arising from the Lasserre hierarchy. In Section 4 we present a basis selection method for the construction of intermediate levels of the Lasserre hierarchy, and provide a method for warm-starting the ADMM. Computational results are presented in Section 5, and the conclusions are provided in Section 6.

1.1 Notation

Given $n, k \in \mathbb{N}$, we define $[n] := \{1, \dots, n\}$ and $\binom{[n]}{\leq k} := \{\beta \subseteq [n] : |\beta| \leq k\}$. Let $\mathcal{B} \subseteq \binom{[n]}{\leq k}$. We define $\mathcal{B}^{2\cup} := \{\beta \cup \beta' : \beta, \beta' \in \mathcal{B}\}$. For any $\beta \subseteq [n]$, we define $\mathbb{1}_\beta \in \{0, 1\}^n$ as the indicator vector corresponding to β , i.e., $(\mathbb{1}_\beta)_i = 1$ if and only if $i \in \beta$. We define $\mathbb{I}(\beta) \in \{0, 1\}$ as the indicator that equals 1 if $|\beta| = 1$, and 0 otherwise. We define \mathbb{R}_+^k as the set of entrywise nonnegative k -dimensional real vectors. We denote by \mathcal{S}^k the set of symmetric matrices of order k , and by \mathcal{S}_+^k the set of symmetric PSD matrices of order k . We write $A \succeq 0$ to indicate that $A \in \mathcal{S}_+^k$. For $a \in \mathbb{R}^k$, we define $\text{Diag}(a) \in \mathcal{S}^k$ as the diagonal matrix with a on its diagonal. For $A, B \in \mathcal{S}^k$, we define the inner product $\langle A, B \rangle := \text{tr}(AB) = \sum_{i=1}^k \sum_{j=1}^k A_{ij} B_{ij}$. The Frobenius norm of $A \in \mathcal{S}^k$ is defined as $\|A\| := \sqrt{\langle A, A \rangle}$. For any closed convex $W \subseteq \mathcal{S}^k$, the projection of $A \in \mathcal{S}^k$ onto W is defined as $\mathcal{P}_W(A) := \arg \min_{X \in W} \|X - A\|$.

Define $\mathbb{R}[x]$ as the set of polynomials in the variables x_1, \dots, x_n with real coefficients. Given $\mathcal{B} \subseteq \binom{[n]}{\leq k}$, we define $\mathbf{x}^{\mathcal{B}}$ as the $|\mathcal{B}|$ -dimensional vector of all monomials $x^{\beta} := \prod_{i \in \beta} x_i$, $\beta \in \mathcal{B}$, with $x^{\emptyset} = 1$. The monomials $(x^{\beta})_{\beta \in \mathcal{B}}$ form a basis of some subspace of $\mathbb{R}[x]$. With slight abuse of terminology, we refer to both $(x^{\beta})_{\beta \in \mathcal{B}}$ and \mathcal{B} as a bases.

2 The Lasserre hierarchy for the stable set problem

We present the Lasserre hierarchy [21] for the stable set problem. Similar derivations can also be found in, e.g., [15, Sect. 3.1], [24], and [25, Example 8.16].

Let $G = (V, E)$ be a simple undirected graph. Without loss of generality, we assume that $V = [n]$ for some $n \in \mathbb{N}$. Let $\beta \subseteq [n]$. If β is a stable set in G , we say that β is stable in G . We define $[n]_G := \{\beta \subseteq [n] : \beta \text{ is stable in } G\}$ as the set of all stable sets of G , and set $S_G := \{\mathbb{1}_{\beta} : \beta \in [n]_G\}$. We define $\mathbb{P}_G \subseteq \mathbb{R}[x]$ as the set of polynomials nonnegative over S_G .

Observe that the stability number $\alpha(G) = \max_{x \in S_G} \sum_{i \in [n]} x_i$, which is equivalent to

$$\alpha(G) = \min_{\mu \in \mathbb{R}} \left\{ \mu : \mu - \sum_{i \in [n]} x_i \in \mathbb{P}_G \right\}.$$

This equivalence follows from the observation that for fixed μ we have $\min_{x \in S_G} \mu - \sum_{i \in [n]} x_i = \mu - \max_{x \in S_G} \sum_{i \in [n]} x_i = \mu - \alpha(G)$, which implies that $\mu - \sum_{i \in [n]} x_i$ is a nonnegative polynomial over S_G if and only if $\mu \geq \alpha(G)$. In general, it is NP-hard to optimize over S_G or \mathbb{P}_G , which motivates the search of tractable relaxations of (2). Let us formulate such a relaxation, in terms of sum of squares (SOS) polynomials. To this end, we define the polynomial ideal

$$\mathcal{I}_G := \langle x_i^2 - x_i \text{ for all } i \in [n], x_i x_j \text{ for all } \{i, j\} \in E \rangle,$$

that is used to define \mathbb{P}_G as follows:

$$\mathbb{P}_G := \left\{ f \in \mathbb{R}[x] : f \equiv \sum_{j \in [k]} f_j^2 \pmod{\mathcal{I}_G}, f_j \in \mathbb{R}[x] \text{ for all } j \in [k], k \in \mathbb{N} \right\},$$

see e.g., [33, Thm. 1]. Note that \mathcal{I}_G encodes the set S_G in the sense that $x \in S_G$ if and only if $f(x) = 0$ for all $f \in \mathcal{I}_G$. Polynomials of the form $\sum_{j \in [k]} f_j^2$ are called SOS polynomials. SOS polynomials, and thus polynomials in \mathbb{P}_G , can be expressed in terms of PSD matrices. Specifically, we have that

$$f \in \mathbb{P}_G \iff f \equiv \left(\mathbf{x}^{\mathcal{B}'} \right)^{\top} A \mathbf{x}^{\mathcal{B}'} \pmod{\mathcal{I}_G} \text{ for some } A \in \mathcal{S}_+^{|\mathcal{B}'|},$$

where $\mathcal{B}' := \binom{[n]}{\leq n}$, see e.g., [22, Prop. 2.1]. This shows that one can optimize over \mathbb{P}_G using semidefinite programming. However, $|\mathcal{B}'|$ is exponential in n , which makes \mathbb{P}_G intractable. It is therefore natural to consider a subset of \mathbb{P}_G by fixing a $\mathcal{B} \subseteq \binom{[n]}{\leq n}$ such that $|\mathcal{B}|$ is polynomial in n . For any $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq n}$, we define a corresponding subset of \mathbb{P}_G as

$$\mathbb{P}_G(\mathcal{B}) := \left\{ f \in \mathbb{R}[x] : f \equiv \left(\mathbf{x}^{\mathcal{B}} \right)^{\top} A \mathbf{x}^{\mathcal{B}} + c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}} \pmod{\mathcal{I}_G}, A \in \mathcal{S}_+^{|\mathcal{B}|}, c \in \mathbb{R}_+^{|\mathcal{B}^{2\cup}|} \right\}.$$

In (2), the term $c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}}$ is introduced to enlarge $\mathbb{P}_G(\mathcal{B})$, resulting in stronger bounds on $\alpha(G)$. Moreover, we show in Section 3.1 that the addition of this term does not increase the computational cost of obtaining these bounds. Note that $c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}}$ is an SOS polynomial modulo \mathcal{I}_G , since $c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}} \equiv \left(\mathbf{x}^{\mathcal{B}^{2\cup}} \right)^{\top} \text{Diag}(c) \mathbf{x}^{\mathcal{B}^{2\cup}} \pmod{\mathcal{I}_G}$, and $\text{Diag}(c) \succeq 0$. Since $\left(\mathbf{x}^{\mathcal{B}} \right)^{\top} A \mathbf{x}^{\mathcal{B}}$ is also an SOS polynomial, it follows that $\mathbb{P}_G(\mathcal{B}) \subseteq \mathbb{P}_G$. From (2), we observe that the value

$$\alpha^{\mathcal{B}}(G) := \min_{\mu \in \mathbb{R}} \left\{ \mu : \mu - \sum_{i \in [n]} x_i \in \mathbb{P}_G(\mathcal{B}) \right\}$$

satisfies $\alpha^{\mathcal{B}}(G) \geq \alpha(G)$, since $\mathbb{P}_G(\mathcal{B}) \subseteq \mathbb{P}_G$. Note that $\alpha^{\mathcal{B}}(G)$ is well-defined since $\binom{[n]}{\leq 1} \subseteq \mathcal{B}$. Computing $\alpha^{\mathcal{B}}(G)$ is equivalent to solving an SDP (see Section 3.1) wherein the PSD variable is of order $|\mathcal{B}|$. If $|\mathcal{B}|$ is polynomial in n , the value $\alpha^{\mathcal{B}}(G)$ can be computed in polynomial time up to fixed precision [35, Cor. 9].

It is worth noting that the computational effort of computing $\alpha^{\mathcal{B}}(G)$ for some $\mathcal{B} \subseteq \binom{[n]}{\leq n}$ can be reduced significantly by computing instead $\alpha^{\mathcal{B} \cap [n]_G}(G)$. Indeed, $|\mathcal{B} \cap [n]_G| \leq |\mathcal{B}|$, which results in a smaller PSD variable, and $\alpha^{\mathcal{B}}(G) = \alpha^{\mathcal{B} \cap [n]_G}(G)$, as the next result shows.

Lemma 1. *Let G be a graph on n vertices and $\mathcal{B} \subseteq \binom{[n]}{\leq n}$. For $\alpha^{\mathcal{B}}(G)$ as in (2), we have $\alpha^{\mathcal{B}}(G) = \alpha^{\mathcal{B} \cap [n]_G}(G)$.*

Proof. For notational convenience, we write $\mathcal{B}' := \mathcal{B} \cap [n]_G$. By the definition of $\alpha^{\mathcal{B}}(G)$, it suffices to show that $\mathbb{P}_G(\mathcal{B}') = \mathbb{P}_G(\mathcal{B})$. Since $\mathcal{B}' \subseteq \mathcal{B}$, it is clear from (2) that $\mathbb{P}_G(\mathcal{B}') \subseteq \mathbb{P}_G(\mathcal{B})$. Thus, it remains to show the reverse inclusion. Let $f \in \mathbb{P}_G(\mathcal{B})$, and let $A \in \mathcal{S}_+^{|\mathcal{B}|}$ and $c \in \mathbb{R}_+^{|\mathcal{B}^{2\cup}|}$ satisfy $f \equiv (\mathbf{x}^{\mathcal{B}})^{\top} A \mathbf{x}^{\mathcal{B}} + c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}} \pmod{\mathcal{I}_G}$, where \mathcal{I}_G is defined in (2). Let A be indexed by the elements of \mathcal{B} , and let A' be the principal submatrix of A indexed by the elements of \mathcal{B}' . We define similarly the vector $c' = (c_{\beta})_{\beta \in (\mathcal{B}')^{2\cup}}$. Since $x^{\beta} \equiv 0 \pmod{\mathcal{I}_G}$ for any $\beta \in \mathcal{B} \setminus \mathcal{B}'$, we have that

$$f \equiv (\mathbf{x}^{\mathcal{B}})^{\top} A \mathbf{x}^{\mathcal{B}} + c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}} \equiv (\mathbf{x}^{\mathcal{B}'})^{\top} A' \mathbf{x}^{\mathcal{B}'} + (c')^{\top} \mathbf{x}^{(\mathcal{B}')^{2\cup}} \pmod{\mathcal{I}_G} \implies f \in \mathbb{P}_G(\mathcal{B}'),$$

which completes the proof. \square

The k th level of the Lasserre hierarchy for the stable set problem is to compute $\alpha^{\binom{[n]}{\leq k}}(G)$, or equivalently, $\alpha^{\binom{[n]}{\leq k} \cap [n]_G}(G)$. For any fixed value of k , $\left| \binom{[n]}{\leq k} \right| \in \mathcal{O}(n^k)$, and thus, $\alpha^{\binom{[n]}{\leq k}}(G)$ can be computed in polynomial time. The sequence $(\alpha^{\binom{[n]}{\leq k}}(G))_{k \in \mathbb{N}}$ is decreasing towards $\alpha(G)$. Moreover, if $\alpha(G) \geq 2$, then $\alpha^{\binom{[n]}{\leq k}}(G) = \alpha(G)$ for $k \geq \alpha(G) - 1$ [23, Prop. 21].

3 The ADMM for computing $\alpha^{\mathcal{B}}(G)$

In this section we propose the ADMM for computing $\alpha^{\mathcal{B}}(G)$, for general bases $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq n}$. The ADMM has been successfully used to solve SDPs, see e.g., [28, 32, 36, 40]. Compared to the IPM, the classical method for solving SDPs, the ADMM requires less memory, making it better suited for solving the large-scale SDPs that arise in the Lasserre hierarchy for the stable set problem.

Throughout the remainder of this section we fix some basis \mathcal{B} that satisfies $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq n}$.

3.1 The SDP defining $\alpha^{\mathcal{B}}(G)$

Consider the constraint $\mu - \sum_{i \in [n]} x_i \in \mathbb{P}_G(\mathcal{B})$ in the definition of $\alpha^{\mathcal{B}}(G)$, given by (2). It follows from (2) that this constraint is equivalent to

$$\mu - \sum_{i \in [n]} x_i \equiv (\mathbf{x}^{\mathcal{B}})^{\top} A \mathbf{x}^{\mathcal{B}} + c^{\top} \mathbf{x}^{\mathcal{B}^{2\cup}} \pmod{\mathcal{I}_G}, \quad A \in \mathcal{S}_+^{|\mathcal{B}|}, \quad c \in \mathbb{R}_+^{|\mathcal{B}^{2\cup}|},$$

where \mathcal{I}_G is defined in (2). Let us index the matrix A with the elements of \mathcal{B} , and c with elements of $\mathcal{B}^{2\cup}$.

The equivalence relation (3.1) implies that the two polynomials have equal coefficients of x^{β} , for all $\beta \in [n]_G$. Thus, (3.1) implies the following linear equalities: $A_{\emptyset, \emptyset} + c_{\emptyset} = \mu$ and

$$\sum_{\beta \in \mathcal{B}} \sum_{\beta' \in \mathcal{B}: \beta \cup \beta' = \gamma} A_{\beta, \beta'} + c_{\gamma} = -\mathbb{I}(\gamma) \quad \text{for all } \gamma \in \mathcal{B}^{2\cup} \cap [n]_G, \quad \gamma \neq \emptyset.$$

Here, we consider $\gamma \in \mathcal{B}^{2\cup}$ since for any $\beta, \beta' \subseteq [n]$, $x^{\beta} x^{\beta'} \equiv x^{\beta \cup \beta'} \pmod{\mathcal{I}_G}$. Since the objective in (2) is to minimize μ , it follows that at optimality, we have $A_{\emptyset, \emptyset} = \mu$ and $c_{\emptyset} = 0$. We eliminate the other entries of c , by transforming the equality constraints from (3.1) into inequality constraints, using that $c \geq 0$. It then follows that

$$\alpha^{\mathcal{B}}(G) = \min \left\{ A_{\emptyset, \emptyset} : A \in \mathcal{S}_+^{|\mathcal{B}|} \cap \mathcal{F}(\mathcal{B}) \right\},$$

where

$$\mathcal{F}(\mathcal{B}) := \left\{ A \in \mathcal{S}^{|\mathcal{B}|} : \sum_{\beta \in \mathcal{B}} \sum_{\beta' \in \mathcal{B}: \beta \cup \beta' = \gamma} A_{\beta, \beta'} \leq -\mathbb{I}(\gamma) \text{ for all } \gamma \in \mathcal{B}^{2\cup} \cap [n]_G, \gamma \neq \emptyset \right\}.$$

We will use formulation (3.1) to compute $\alpha^{\mathcal{B}}(G)$ via the ADMM.

3.2 The ADMM iterates

To apply the ADMM to (3.1), we first reformulate (3.1) as the following optimization problem:

$$\begin{aligned} \min_{X, Y \in \mathcal{S}^{|\mathcal{B}|}} \quad & Y_{\emptyset, \emptyset} \\ \text{s.t.} \quad & X \in \mathcal{S}_+^{|\mathcal{B}|}, Y \in \mathcal{F}(\mathcal{B}), X = Y, \end{aligned}$$

where $\mathcal{F}(\mathcal{B})$ is defined in (3.1). Given a penalty parameter $\rho > 0$, the augmented Lagrangian associated to (3.2), with respect to the constraint $X = Y$, is the function

$$L_\rho(X, Y, Z) := Y_{\emptyset, \emptyset} + \rho \langle Z, Y - X \rangle + \frac{\rho}{2} \|Y - X\|^2,$$

where $Z \in \mathcal{S}^{|\mathcal{B}|}$ is the (scaled) dual variable. Given some initial $X^1, Y^1, Z^1 \in \mathcal{S}^{|\mathcal{B}|}$, the ADMM computes the sequence of matrices $(X^\ell, Y^\ell, Z^\ell)_{\ell \in \mathbb{N}}$, defined recursively as

$$\begin{aligned} X^{\ell+1} &:= \arg \min_{X \succeq 0} L_\rho(X, Y^\ell, Z^\ell) \\ Y^{\ell+1} &:= \arg \min_{Y \in \mathcal{F}(\mathcal{B})} L_\rho(X^{\ell+1}, Y, Z^\ell) \\ Z^{\ell+1} &:= Z^\ell + \nu (Y^{\ell+1} - X^{\ell+1}), \end{aligned}$$

where $\nu \in \mathbb{R}$ is a stepsize parameter. It is known that the matrices X^ℓ and Y^ℓ converge with rate $\mathcal{O}(1/\ell)$ (in the ergodic sense) to an optimal solution of (3.2) [17, Thm. 6.5] when $\nu \in \left(0, \frac{1+\sqrt{5}}{2}\right)$ [11, Thm. 5.1].

The minimization problems in (3.2) admit the following closed form solutions, see e.g., [32, Eq. 3.4]:

$$\begin{aligned} \arg \min_{X \succeq 0} L_\rho(X, Y^\ell, Z^\ell) &= \mathcal{P}_{\mathcal{S}_+^{|\mathcal{B}|}}(Y^\ell + Z^\ell), \\ \arg \min_{Y \in \mathcal{F}(\mathcal{B})} L_\rho(X^{\ell+1}, Y, Z^\ell) &= \mathcal{P}_{\mathcal{F}(\mathcal{B})}\left(X^{\ell+1} - \frac{1}{\rho}H - Z^\ell\right), \end{aligned}$$

where $H \in \mathcal{S}^{|\mathcal{B}|}$ is the matrix that is zero everywhere, except for the entry $H_{\emptyset, \emptyset} = 1$. Note that $\langle H, X \rangle = X_{\emptyset, \emptyset}$. The augmented Lagrangian (3.2) and scheme (3.2) correspond to the *scaled form* of the ADMM, see e.g., [5, Sect. 3.1.1]. Compared to the unscaled form, the scaled form of the ADMM avoids the multiplication of $(1/\rho)$ with Z^ℓ , see (3.2).

We briefly discuss the two projections in (3.2). For any $A \in \mathcal{S}^{|\mathcal{B}|}$,

$$\mathcal{P}_{\mathcal{S}_+^{|\mathcal{B}|}}(A) = \sum_{\lambda \in \Lambda(A): \lambda > 0} \lambda u_\lambda u_\lambda^\top,$$

where $\Lambda(A)$ denotes the eigenspectrum of A , and the vectors $(u_\lambda)_{\lambda \in \Lambda(A)}$ form an orthonormal basis of eigenvectors. To project a symmetric matrix Y onto $\mathcal{F}(\mathcal{B})$, see (3.1), we consider $\mathcal{F}(\mathcal{B})$ as a set of vectors by identifying Y with its upper triangular entries. To account for the symmetry of Y , we replace the terms $Y_{\beta, \beta'} + Y_{\beta', \beta}$ with $2Y_{\beta, \beta'}$. From this point of view, it can be seen that $\mathcal{F}(\mathcal{B})$ is (up to reordering) a Cartesian product of closed half-spaces, one for each nonempty $\gamma \in \mathcal{B}^{2\cup} \cap [n]_G$. Therefore, projecting onto $\mathcal{F}(\mathcal{B})$ is equivalent to projecting onto each half-space separately. Consider such a half-space corresponding to some γ , defined by $a^\top x \leq -\mathbb{I}(\gamma)$, where $a \in \{1, 2\}^p$, for some $p \in \mathbb{N}$. The projection of some $z \in \mathbb{R}^p$ onto this half-space is given by

$$\arg \min_{x \in \mathbb{R}^p: a^\top x \leq -\mathbb{I}(\gamma)} (x - z)^\top \text{Diag}(a)(x - z).$$

The presence of $\text{Diag}(a)$ in the objective function ensures that off-diagonal elements of Y are weighted with a factor of 2, as they appear twice in Y . The following result shows that (3.2) can be expressed in closed form.

Lemma 2. Let $p \in \mathbb{N}$, $a, z \in \mathbb{R}^p$ with $a > 0$, $b \in \mathbb{R}$, and denote by $\mathbf{1}_p \in \mathbb{R}^p$ the all-ones vector. We have that

$$\arg \min_{x \in \mathbb{R}^p: a^\top x \leq b} (x - z)^\top \text{Diag}(a)(x - z) = z - \frac{\max\{a^\top z - b, 0\}}{\mathbf{1}_p^\top a} \mathbf{1}_p.$$

Proof. Let $f(x) := (x - z)^\top \text{Diag}(a)(x - z)$. We need to determine the minimizer of the problem

$$\min_{x \in \mathbb{R}^p} f(x) \text{ subject to } a^\top x \leq b.$$

We consider two cases. If $a^\top z \leq b$, then z minimizes (3.2), since for any $x \in \mathbb{R}^p$, we have $0 = f(z) \leq f(x)$. Here, the inequality follows from the fact that $a > 0$, which makes $\text{Diag}(a)$ positive definite.

If $a^\top z > b$, we consider the Karush-Kuhn-Tucker (KKT) conditions [19, 20] corresponding to (3.2), which state the following: if (x^*, λ^*) , with $\lambda^* \geq 0$, is a saddle point of the Lagrangian $L(x, \lambda) := f(x) + \lambda(a^\top x - b)$, then x^* minimizes (3.2). The saddle point of $L(x, \lambda)$ is computed as follows:

$$\frac{\partial L(x, \lambda)}{\partial x} = 2 \text{Diag}(a)(x - z) + \lambda a = 0 \implies x^* = z - \frac{\lambda^*}{2} \mathbf{1}_p.$$

Solving $\partial L(x^*, \lambda)/\partial \lambda = a^\top x^* - b = a^\top(z - \frac{\lambda^*}{2} \mathbf{1}_p) - b = 0$ for λ^* yields $\lambda^* = 2(a^\top z - b)/(\mathbf{1}_p^\top a) \geq 0$, where the inequality follows from $a^\top z > b$. Then $x^* = z - \frac{a^\top z - b}{\mathbf{1}_p^\top a} \mathbf{1}_p$ minimizes (3.2) by the KKT conditions.

The proof follows by combining the two cases into the form (2). \square

3.3 Valid bounds on $\alpha(G)$ from the ADMM iterates

The following lemma shows that any matrix in $\mathcal{S}_+^{|\mathcal{B}|}$ induces an upper bound on $\alpha(G)$. Note that the matrix X^ℓ from the ADMM iterates (3.2) satisfies $X^\ell \in \mathcal{S}_+^{|\mathcal{B}|}$ for all $\ell \in \mathbb{N}$. Thus, any iteration of the ADMM provides an upper bound on $\alpha(G)$.

Lemma 3. Let $G = (V, E)$, where $V = [n]$ for some $n \in \mathbb{N}$, $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq n}$, $M \in \mathcal{S}_+^{|\mathcal{B}|}$, and let $(f_\beta)_{\beta \in \mathcal{B}^{2\cup} \cap [n]_G}$ satisfy $(\mathbf{x}^\mathcal{B})^\top M \mathbf{x}^\mathcal{B} \equiv \sum_{\beta \in \mathcal{B}^{2\cup} \cap [n]_G} f_\beta x^\beta \pmod{\mathcal{I}_G}$. We have that the value

$$v(M) := M_{\emptyset, \emptyset} + \sum_{\beta \in \mathcal{B}^{2\cup} \cap [n]_G: \beta \neq \emptyset} \max\{f_\beta + \mathbb{I}(\beta), 0\}$$

satisfies $v(M) \geq \alpha(G)$.

Proof. For notational convenience, we define $\mathcal{B}' := \mathcal{B}^{2\cup} \cap [n]_G$. Consider the polynomial

$$g(x) := (\mathbf{x}^\mathcal{B})^\top M \mathbf{x}^\mathcal{B} + \sum_{\beta \in \mathcal{B}': \beta \neq \emptyset} \max\{f_\beta + \mathbb{I}(\beta), 0\} (1 - x^\beta) + \sum_{\beta \in \mathcal{B}': \beta \neq \emptyset} \max\{-f_\beta - \mathbb{I}(\beta), 0\} x^\beta.$$

Observe that for any $x \in S_G$, $g(x) \geq 0$ since $M \succeq 0$, and $1 - x^\beta, x^\beta \geq 0$. Hence, $g \in \mathbb{P}_G$.

Let $(g_\beta)_{\beta \in \mathcal{B}'}$ be the coefficients of g , i.e., $g \equiv \sum_{\beta \in \mathcal{B}'} g_\beta x^\beta \pmod{\mathcal{I}_G}$. For nonempty $\beta \in \mathcal{B}'$, we have

$$g_\beta = f_\beta - \max\{f_\beta + \mathbb{I}(\beta), 0\} + \max\{-f_\beta - \mathbb{I}(\beta), 0\} = -\mathbb{I}(\beta).$$

Thus, $g(x) \equiv g_\emptyset - \sum_{i \in [n]} x_i \pmod{\mathcal{I}_G}$, where g_\emptyset is the constant term of g , given by $g_\emptyset = M_{\emptyset, \emptyset} + \sum_{\beta \in \mathcal{B}': \beta \neq \emptyset} \max\{f_\beta + \mathbb{I}(\beta), 0\}$. By (2) and the fact that $g \in \mathbb{P}_G$, it follows that $g_\emptyset \geq \alpha(G)$. Since $g_\emptyset = v(M)$, this proves the claim. \square

Thus, it follows from Lemma 3 that for X^ℓ as in (3.2) and v as in (3), $v(X^\ell)$ is an upper bound on the stability number.

4 A dynamic basis selection method and ADMM initialization

We provide a method for selecting a basis $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq 2}$ for computing $\alpha^\mathcal{B}(G)$, see (2). We also provide a method for initializing the ADMM iterates X^1, Y^1, Z^1 , see (3.2). Both these methods are

based on an SDP defining the Lovász theta function of a graph $G = ([n], E)$, with $n \in \mathbb{N}$. This SDP is given by:

$$\begin{aligned} \vartheta(G) = \max_{Z \in \mathcal{S}_+^{1+n}} \quad & \sum_{i \in [n]} Z_{\emptyset, i} \\ \text{s.t.} \quad & Z_{i, j} = 0 \text{ for all } \{i, j\} \in E \\ & Z_{i, i} = Z_{\emptyset, i} \text{ for all } i \in [n], \quad Z_{\emptyset, \emptyset} = 1. \end{aligned}$$

Here, the matrix Z is indexed by the $1 + n$ elements of $\binom{[n]}{\leq 1}$ (see also [15, Sect. 3.1]). It can be shown that (4) is dual to the SDP defining $\alpha^{(\binom{[n]}{\leq 1})}(G)$, see (3.1). To solve (4), we use the IPM-based SDP solver MOSEK [30]. The PSD variable in (4) is of order $1 + n$ and there are $2|E| + 2n + 1$ linear equality constraints, which is small enough for IPM-based solvers to be efficient for graphs of sizes considered here. For instance, among the graphs we consider, `c_fat200_5` attains the largest value of $2|E| + 2n + 1$, with $|E| = 11427$ and $n = 200$ (see Table 7). For this graph, MOSEK solves (4) in approximately 35 seconds.

4.1 The basis selection method

Given a graph G on n vertices, and a maximum basis size $s \geq 1 + n$, our method aims to select a basis $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq 2}$, $|\mathcal{B}| \leq s$ that minimizes $\alpha^{\mathcal{B}}(G)$. The inclusions $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq 2}$ can be interpreted in terms of $\mathcal{F}(\mathcal{B})$, see (3.1), as follows: matrices in $\mathcal{F}\left(\binom{[n]}{\leq 1}\right)$ are submatrices of matrices in $\mathcal{F}(\mathcal{B})$, which in turn are submatrices of matrices in $\mathcal{F}\left(\binom{[n]}{\leq 2}\right)$.

Recall that the first and second levels of the Lasserre hierarchy for the stable set problem are the SDPs defining $\alpha^{(\binom{[n]}{\leq 1})}(G)$ and $\alpha^{(\binom{[n]}{\leq 2})}(G)$ respectively. Thus, our method selects a basis \mathcal{B} such that $\alpha^{\mathcal{B}}(G)$ corresponds to a level of the Lasserre hierarchy intermediate to levels 1 and 2, and for some smaller graphs, equal to level 2. We do not consider bases corresponding to levels $k > 2$, since the value $\alpha^{(\binom{[n]}{\leq 2})}(G)$, after rounding down to the nearest integer, often closes the gap with $\alpha(G)$, or is intractable to compute.

To explain our method, we define, for $\beta \subseteq [n]$, the binary matrix $Z^\beta := \begin{bmatrix} 1 \\ \mathbb{1}_\beta \end{bmatrix} \begin{bmatrix} 1 \\ \mathbb{1}_\beta \end{bmatrix}^\top$, indexed by the $1 + n$ elements of $\binom{[n]}{\leq 1}$. Observe that Z^β is feasible for (4) if and only if $\beta \in [n]_G$. Let $\beta \in [n]_G$ correspond to a maximum stable set, and consider the binary values $(Z_{i, j}^\beta)_{\{i, j\} \notin E}$. Observe that $Z_{i, j}^\beta = 1$ if and only if both vertices $i, j \in \beta$. Thus, the values 1 in the vector $(Z_{i, j}^\beta)_{\{i, j\} \notin E}$ indicate the maximum stable set β . As such, we would like to include the monomials $x_i x_j$, for which $Z_{i, j}^\beta = 1$, in our basis. In practice however, the matrix $Z_{i, j}^\beta$ is unknown, since a maximum stable set is not known. Therefore, instead of Z^β , we consider matrix Z^* , which denotes an optimal solution of (4), and can be considered a semidefinite approximation of Z^β . Then, we include monomial $x_i x_j$, $\{i, j\} \notin E$, in our basis if $Z_{i, j}^*$ is ‘large enough’.

We now present our basis selection method formally: Compute first s' , defined as the cardinality of $\binom{[n]}{\leq 2} \setminus \left(\binom{[n]}{\leq 1} \cup E\right)$, which is the set of non-edges in G . Then we distinguish two cases, based on the given maximum basis size s .

Case 1: $s < 1 + n + s'$. Solve (4) using an IPM-based SDP solver to obtain an optimal solution Z^* . As basis \mathcal{B} , we choose the sets in $\binom{[n]}{\leq 1}$ and the $s - (1 + n)$ non-edges $\{i, j\}$ with largest value $Z_{i, j}^*$. Note that $|\mathcal{B}| = \left|\binom{[n]}{\leq 1}\right| + s - (1 + n) = s$.

Case 2: $s \geq 1 + n + s'$. We take the basis $\mathcal{B} = \binom{[n]}{\leq 2} \setminus E$, of size $|\mathcal{B}| = 1 + n + s' \leq s$. Observe that $\mathcal{B} = \binom{[n]}{\leq 2} \cap [n]_G$, which implies that $\alpha^{\mathcal{B}}(G) = \alpha^{(\binom{[n]}{\leq 2})}(G)$, see Lemma 1. Therefore, in Case 2, the resulting basis corresponds to the second level of the Lasserre hierarchy.

Preliminary numerical experiments showed that selecting monomials based on the largest values of $(Z_{i, j}^*)_{\{i, j\} \notin E}$, outperformed methods that selected monomials based on smallest values in $(Z_{i, j}^*)_{\{i, j\} \notin E}$, or those values closest to the average $\sum_{\{i, j\} \in E} Z_{i, j}^* / |E|$. We also tested basis selection methods based on the values $(Z_{\emptyset, i}^*)_{i \in [n]}$, or degrees of vertices, and these were also outperformed by the above described basis selection method.

4.2 Initialization of ADMM iterates

Our initialization method is defined for any basis $\binom{[n]}{\leq 1} \subseteq \mathcal{B} \subseteq \binom{[n]}{\leq n}$, and depends on optimal primal and dual solutions to (4), denoted respectively by $Z^* \in \mathcal{S}_+^{1+n}$ and $X^* \in \mathcal{S}_+^{1+n}$.

Note that the initial ADMM iterates X^1, Y^1, Z^1 are indexed by the elements of \mathcal{B} . Our initialization method sets the principal submatrix of X^1 , that is indexed by the elements of $\binom{[n]}{\leq 1}$, equal to X^* . We set the other elements of X^1 to zero, and set $Y^1 = X^1$. We initialize Z^1 by setting the principal submatrix of Z^1 corresponding to the elements of $\binom{[n]}{\leq 1}$ as $(1/\rho)Z^*$, and the rest all zero. Here, we scale Z^* by $1/\rho$, since (3.2) corresponds to the scaled form of the ADMM.

An important property of this initialization is that $v(X^1)$, see (3), equals $\vartheta(G)$. Indeed, since $X^* \succeq 0$, also $X^1 \succeq 0$. Moreover, since X^* is a feasible dual solution to (4), and (4) is the SDP dual of (3.1) for $\mathcal{B} = \binom{[n]}{\leq 1}$, it follows that $X^* \in \mathcal{F}\left(\binom{[n]}{\leq 1}\right)$. This implies that the values f_β in (3), corresponding to X^1 , satisfy $f_\beta \leq -\mathbb{I}(\beta)$, from where it follows that $v(X^1) = X_{\emptyset, \emptyset}^1$. Lastly, since X^* is an optimal dual solution to (4), $X_{\emptyset, \emptyset}^* = X_{\emptyset, \emptyset}^1 = \vartheta(G)$. Thus, $v(X^1) = \vartheta(G)$. Consequently, the best bound returned by the ADMM after a finite number of iterations is at most $\vartheta(G)$.

5 Computational results

We compute Lasserre hierarchy bounds for the stable set problem, using the ADMM as described in Section 3.2, with stepsize parameter $\nu = 3/2$, see (3.2). We set the ADMM penalty parameter $\rho = (4/5)\sqrt{|\mathcal{B}|}$, see (3.2), where \mathcal{B} is the used basis of size at most $s \in \mathbb{N}$, determined by the basis selection method outlined in Section 4.1. The algorithms are implemented in MATLAB. All experiments are run on a machine with 16GB RAM and an Intel i7-1165G7 CPU.

In Section 5.1 we compare SDP bounds obtained from two versions of the ADMM algorithm: one that computes the eigenvalue decomposition in (3.2) using single precision, and another that uses double precision. We conclude that the single precision ADMM requires less computation time per iteration, without a significant loss in quality of the corresponding bounds. The stopping conditions are provided in Section 5.2, and are used in Section 5.3 to compute bounds on $\alpha(G)$ from the Lasserre hierarchy with the single precision ADMM. We compare these bounds with the bounds obtained by the exact subgraph hierarchy (ESH) [13] and the bounds from [34]. Both approaches provide among the strongest SDP bounds for the stable set problem.

5.1 Single vs. double precision for eigenvalue decompositions

It is well-known that one of the most expensive steps of the ADMM, when applied to SDP, is computing projections onto the PSD cone, see e.g., [32, Sect. 5] and [36, Sect. 1]. It is standard to compute these projections $\mathcal{P}_{\mathcal{S}_+^{|\mathcal{B}|}}(A)$, see (3.2), by computing the full eigenvalue decomposition of A , that is, $A = \sum_{\lambda \in \Lambda(A)} \lambda u_\lambda u_\lambda^\top$. Computing the eigenvalue decomposition in single precision is computationally less expensive than in double precision. However, the lower accuracy of single precision might result in worse upper bounds compared to double precision. We investigate this trade-off by comparing the SDP bounds on the stable set problem obtained by two versions of the ADMM algorithm: one that uses single precision and another that uses double precision for the eigenvalue decompositions. All other parts of the algorithms remain the same.

We run each version of the ADMM algorithm for a fixed number of iterations to compute bounds on $\alpha(G)$ for three different graphs. All ADMM iterates, see (3.2), are initialized by setting $X^1 = Y^1 = Z^1 = 0$, i.e., the zero matrix of appropriate size. When the iteration number ℓ is a multiple of 100, we compute the bound $v(X^\ell)$, see (3), which satisfies $v(X^\ell) \geq \alpha(G)$. We also track the computation time, and present the results in Tables 1 to 3. In these tables, column ‘Bnd.diff.’ (for bound difference) reports the double precision bound $v(X^\ell)$ subtracted from the single precision $v(X^\ell)$.

From the data presented in these tables we conclude that the difference in $v(X^\ell)$ between the two precisions and for fixed ℓ , is negligible (at most 0.04517). In contrast, the reduction in computation time may be significant, see for instance the row corresponding to $\ell = 600$ in Table 3: the single precision ADMM requires 113.82 seconds to compute 600 iterations, whereas the double precision ADMM requires 205.35 seconds. With this larger computation time, the double precision bound $v(X^\ell)$ is only 0.00677 smaller than the single precision bound. Rounded down, both precisions provide the same bound on

$\alpha(G)$, but the single precision ADMM requires only 55% of the computation time of the double precision ADMM.

We also perform the following similar experiment: we run each ADMM version for one hour on a single graph. We pick a basis of size $s = 2500$ and initialize the ADMM iterates with the methods outlined in Sections 4.1 and 4.2, respectively. At the first iteration and every 10 seconds t , we compute the valid upper bound $v_t := v(X^\ell)$, where ℓ is the iteration index at time t . We set $v_0 := v(X^1) = \vartheta(G)$. This yields a set of upper bounds $\{v_0, v_{10}, \dots, v_{3600}\}$. Figure 1 reports the best bounds achieved by both ADMM versions over time. That is, Figure 1 plots the curves through the points $(t, \min_{k \in \{0, 10, \dots, t\}} v_k)$, for $t \in \{0, 10, 20, \dots, 3600\}$. For the single precision version of the ADMM algorithm, the bounds $v_{10}, v_{20}, \dots, v_{80} \geq v_0$, and thus the plot corresponding to single precision is flat for the first 80 seconds. For the double precision variant of the ADMM, the plot is flat for the first 170 seconds. Figure 1 demonstrates that the single precision ADMM provides stronger bounds than the double precision ADMM, at any time t , $t \leq 3600$. This is due to the fact that the single precision ADMM algorithm can perform more ADMM iterations in the same time as compared to the double precision ADMM algorithm.

Based on these conclusions, we will use the version of the ADMM that computes eigenvalue decompositions in single precision for the remainder of this section.

5.2 Stopping conditions

For each graph, we run the ADMM algorithm for at most one hour (unless otherwise specified). Next to maximum running time, we have the following stopping conditions: we stop if

$$\max \left\{ \frac{\|X^\ell - Y^\ell\|}{1 + \|X^\ell\|}, \rho \frac{\|X^{\ell-1} - X^\ell\|}{1 + \|X^\ell\|} \right\} \leq 10^{-4},$$

for 3 consecutive iterations, see e.g., [5, Sect. 3.3.1]. To keep computation costs minimal, we only verify whether (5.2) holds whenever the iteration number ℓ is a multiple of 100 (and then also for the consecutive iterations if (5.2) holds). We also stop earlier if the objective value $X_{\emptyset, \emptyset}^\ell$ stagnates. Specifically, we stop if

$$|X_{\emptyset, \emptyset}^\ell - X_{\emptyset, \emptyset}^{\ell-1}| < 10^{-5}$$

for $K_{\text{stag}} \in \mathbb{N}$ (not necessarily consecutive) iterations. For all the tables that follow, except Table 5, we set $K_{\text{stag}} = 150$.

ℓ	Single precision		Double precision		Bnd. diff.
	$v(X^\ell)$	Time (s)	$v(X^\ell)$	Time (s)	
100	7.13476	0.14	7.13484	0.16	-0.00008
200	7.03090	0.25	7.03090	0.32	0.00000
300	7.07006	0.36	7.07013	0.47	-0.00007
400	7.00445	0.46	7.00444	0.61	0.00000
500	7.00680	0.57	7.00677	0.76	0.00004
600	7.00032	0.68	7.00032	0.91	0.00001

Table 1: Comparison of upper bounds for HoG_15599, $n = 20$, $\alpha(G) = 7$, $s = 126$.

5.3 SDP-Lasserre bounds on $\alpha(G)$

We investigate the quality of the upper bounds on the stability number of graphs obtained from the Lasserre hierarchy, and computed via the ADMM. For the remainder of this section, we refer to such bounds as SDP-Lasserre bounds. For all the tables that follow, except Table 5, we set the maximum basis size $s = 2500$. This value is chosen to ensure the ADMM converges within one hour on most graphs. In the remainder of this section, we initialize the ADMM iterates using the method described in Section 4.2.

5.3.1 SDP bounds for evil, random and near-regular graphs

We benchmark the SDP-Lasserre bounds on the complement of evil graphs¹ [41], as well as on random graphs and near-regular graphs. These graphs are also tested in [34], and can be described as follows:

¹The evil graphs are available at <https://github.com/zbogdan/evil2/tree/main/evil-tests>.

ℓ	Single precision		Double precision		Bnd. diff.
	$v(X^\ell)$	Time (s)	$v(X^\ell)$	Time (s)	
100	20.92761	5.14	20.92756	7.99	0.00005
200	23.01029	9.97	23.01063	16.05	-0.00035
300	17.21289	15.28	17.21270	24.45	0.00019
400	16.42331	21.03	16.42139	33.03	0.00192
500	16.51678	26.66	16.51551	41.41	0.00127
600	16.30446	32.15	16.30274	50.83	0.00172
700	16.49339	37.60	16.49267	59.29	0.00072
800	16.29119	43.19	16.28875	67.88	0.00245
900	16.39126	49.01	16.38704	76.83	0.00421
1000	16.28313	54.91	16.27664	85.77	0.00649
1100	16.31098	62.01	16.30117	94.50	0.00981

Table 2: Comparison of upper bounds for MANN_a9_clq, $n = 45$, $\alpha(G) = 16$, $s = 964$.

ℓ	Single precision		Double precision		Bnd. diff.
	$v(X^\ell)$	Time (s)	$v(X^\ell)$	Time (s)	
100	122.53210	19.66	122.52626	35.47	0.00584
200	82.37548	41.58	82.37179	70.49	0.00368
300	78.23423	59.13	78.22998	105.84	0.00425
400	78.89802	76.84	78.86581	139.59	0.03221
500	71.93651	96.10	71.89134	173.73	0.04517
600	69.47637	113.82	69.46960	205.35	0.00677

Table 3: Comparison of upper bounds for evil-N150-p98-myc5x30, $n = 150$, $\alpha(G) = 60$, $s = 1500$.

- **Evil graphs [41].** These are benchmark graphs for the NP-hard clique problem. The clique problem on a graph G is equivalent to the stable set problem on the complement graph of G . Therefore, we consider the complement of evil graphs.
- **Random graphs.** These are generated following the Erdős-Rényi model. That is, given $n \in \mathbb{N}$ and $p \in (0, 1)$, generate a graph by taking n vertices and creating edges $\{i, j\}$ independently with probability p for every $i, j \in [n]$.
- **Near-regular graphs.** For given $n, r \in \mathbb{N}$ such that nr is even, these graphs are constructed as follows (see also [13, Sect. 7.2]): consider a set of nr vertices given by $\tilde{V} := \{\{i, b\} : i \in [n], b \in [r]\}$. Select a perfect matching on the vertices in \tilde{V} to obtain the edge set $\tilde{E} \subseteq \tilde{V} \times \tilde{V}$. Consider now the graph G with vertices $V = [n]$ and edge set $E = \{\{i, j\} : \exists b, b' \in [r] \text{ s.t. } \{\{i, b\}, \{j, b'\}\} \in \tilde{E}\}$. Note that G is a regular multigraph. Remove from G any parallel edges and self-loops. The resulting graph is said to be near-regular.

We use the same exact random and near-regular graphs as in [34]. For each graph, we use the method provided in Section 4.1 to select a basis of size 2500 for computing the SDP-Lasserre bounds. Because $2500 < \left| \binom{[n]}{\leq 2} \cap [n]_G \right|$ for all graphs, the resulting bounds correspond to the Lasserre hierarchy at levels intermediate to 1 and 2.

Table 4 reports the SDP-Lasserre bounds on $\alpha(G)$, computed via the ADMM, in the column ‘SDP-Lasserre’. Columns n , $|E|$, and $\vartheta(G)$ report the number of vertices, number of edges, and Lovász theta function respectively, for each graph. Column $\alpha(G)$ reports (bounds on) the stability number of the graphs. For evil graphs, the value of $\alpha(G)$ is known by construction. The values and intervals for the stability numbers of the random and near-regular graphs are taken from [14, Table 6]. In Table 5, we present improved bounds on $\alpha(G)$ compared to [14, Table 6].

The columns under ‘BOUND 2 [34]’ in Table 4 report BOUND 2 from the recent paper [34, Sect. 4.1], which provides one of the best SDP-based upper bounds on $\alpha(G)$. BOUND 2 is obtained by strengthening the Lovász theta function with additional valid inequalities, such as triangle inequalities and inequalities induced by complete subgraphs of G (i.e., constraints of the form $\sum_{i \in U} x_i \leq 1$ where U is a set of pairwise connected vertices). BOUND 2 is computed by the IPM-based SDP solver MOSEK [30]. We

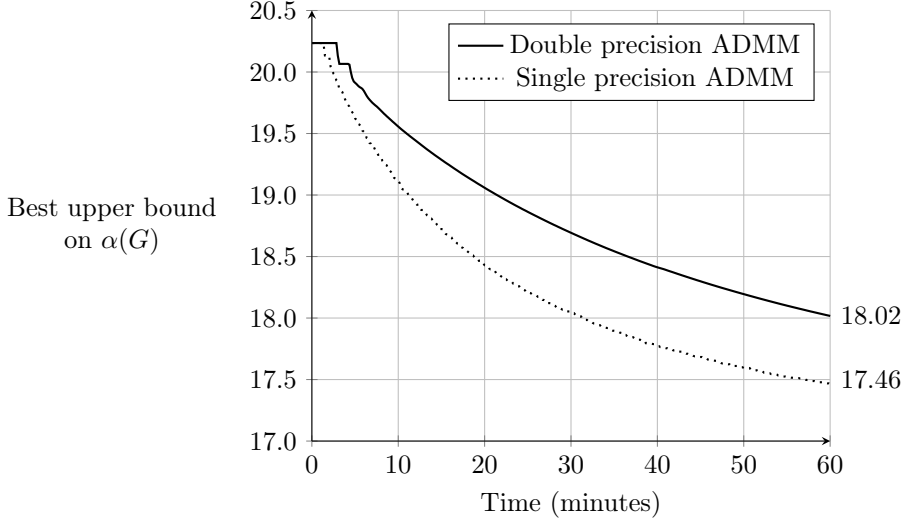


Figure 1: Comparison of upper bounds for `evil-N184-p98-myc23x8`, $n = 184$, $\alpha(G) = 16$, $s = 2500$.

computed BOUND 2 on the same machine we used to compute the SDP-Lasserre bounds². Additional details regarding BOUND 2 can be found in [34, Sect. 4.1].

The columns ‘GapClsd’ (for gap closed) under SDP-Lasserre and BOUND 2 report the fraction $\frac{\vartheta(G) - f^*}{\vartheta(G) - \alpha(G)}$ rounded down to the first digit, where f^* equals the value of the corresponding bound. Note that $f^* \in [\alpha(G), \vartheta(G)]$. For the three graphs with unknown $\alpha(G)$, we use the best known lower bound on $\alpha(G)$ from [14, Table 6] to compute GapClsd. Lastly, bounds that equal $\alpha(G)$ when rounded down are boldfaced.

Computing BOUND 2 for the graphs in Table 4 requires on average only 4 minutes of computation time per graph. The ADMM required on average 39 minutes per graph. The SDP-Lasserre bounds improve over BOUND 2 for various graphs, sometimes closing the gap towards $\alpha(G)$ whereas BOUND 2 did not (see the graphs `evil-N125-p98-s3m25x5`, `evil-N138-p98-myc23x6`, `reg_n100_r6` and `reg_n100_r8`). There is one graph in Table 4, `rand_n200_p002`, for which the SDP-Lasserre bound does not improve upon the Lovász theta function $\vartheta(G) = \alpha^{(\lfloor n \rfloor)}(G) = 95.778$. We ran our ADMM algorithm for 4 hours to recompute the SDP-Lasserre bound for this graph, which resulted in an improved bound of 95.244, see also Table 8.

We rerun the ADMM algorithm on the graphs `reg_n200_r10` and `reg_n200_r6` with increased basis size $|\mathcal{B}|$ and extended running time. These changes yield improved upper bounds on $\alpha(G)$ compared to those in [14, Table 6], see also Table 4. The improved bounds are reported in Table 5, where column ‘ $|\mathcal{B}|$ ’ reports the used basis size and column ‘T. (min.)’ reports the runtime rounded to the nearest minute. In these computations, we use $K_{\text{stag}} = 500$, see (5.2).

Lastly, we compare the SDP-Lasserre bounds also with the ESH approach from [13]. Specifically, we compare the SDP-Lasserre bounds with the ESH bounds on the random and near-regular graphs reported in Tables 4 and 5 of [13]. Computational results are presented in Table 6, with identical columns as Table 4, and the newly added column ‘ESH’. The columns under ‘ESH’ report the obtained bounds by the ESH and corresponding GapClsd values. The SDP-Lasserre bounds improve over the ESH bounds in 11 of the 15 reported graphs. The computation times of these ESH bounds is reported in [13], although the used computer is not specified. For each of the graphs in Table 6, the ESH required on average 1454 seconds. Note that SDP-Lasserre bounds, when rounded down to the closest integer, close the gap for more graphs than the other two approaches.

5.3.2 SDP bounds on graphs from [12]

We investigate the quality of the SDP-Lasserre bounds on the graphs tested in [12]. This set of graphs³ contains, among others, complements of DIMACS graphs, additional random Erdős-Rényi graphs, a spin glass graph, graphs from [6], as well as a Paley, a circulant and a cubic graph. Note that stability

²The code for BOUND 2 is available at <https://arxiv.org/src/2401.17069v2/anc>.

³The graphs used in [12] are available at https://arxiv.org/src/2003.13605v6/anc/Data_InputGraphs.mat.

Graph name	n	$ E $	$\alpha(G)$	SDP-Lasserre		BOUND 2 [34]		$\vartheta(G)$
				Bound	GapClsd	Bound	GapClsd	
evil-N120-p98-chv12x10	120	545	20	20.355	92.1%	20.000	99.9%	24.526
evil-N120-p98-myc5x24	120	236	48	48.466	89.8%	48.000	99.9%	52.607
evil-N121-p98-myc11x11	121	508	22	22.361	91.7%	22.000	99.9%	26.397
evil-N125-p98-s3m25x5	125	873	20	20.291	94.1%	22.361	52.7%	25.000
evil-N138-p98-myc23x6	138	1242	12	12.501	84.2%	15.177	0.0%	15.177
evil-N150-p98-myc5x30	150	338	60	61.477	71.1%	60.000	99.9%	65.121
evil-N150-p98-s3m25x6	150	1102	24	25.239	79.3%	26.833	52.7%	30.000
evil-N154-p98-myc11x14	154	701	28	28.316	94.3%	28.000	99.9%	33.596
evil-N180-p98-chv12x15	180	944	30	30.391	94.2%	30.000	99.9%	36.788
evil-N180-p98-myc5x36	180	439	72	75.626	29.0%	72.000	99.9%	77.110
evil-N184-p98-myc23x8	184	1764	16	17.504	64.4%	20.235	0.0%	20.235
evil-N187-p98-myc11x17	187	901	34	34.332	95.1%	34.000	99.9%	40.795
evil-N200-p98-s3m25x8	200	1550	32	35.979	50.2%	35.777	52.7%	40.000
evil-N210-p98-myc5x42	210	541	84	86.513	58.1%	84.000	99.9%	90.001
evil-N220-p98-myc11x20	220	1130	40	42.791	65.0%	40.000	99.9%	47.994
evil-N230-p98-myc23x10	230	2263	20	22.553	51.7%	25.294	0.0%	25.294
evil-N240-p98-chv12x20	240	1352	40	40.461	94.9%	40.000	99.9%	49.051
evil-N240-p98-myc5x48	240	718	96	98.627	44.0%	96.011	99.7%	100.696
evil-N250-p98-s3m25x10	250	2050	40	45.003	49.9%	44.721	52.7%	50.000
evil-N253-p98-myc11x23	253	1456	46	49.447	62.5%	46.014	99.8%	55.193
evil-N300-p98-myc5x60	300	1033	120	121.876	34.4%	120.020	99.2%	122.861
rand_n100_p004	100	212	45	45.140	86.8%	45.027	97.4%	46.067
rand_n100_p006	100	303	38	38.199	91.5%	38.435	81.5%	40.361
rand_n100_p008	100	443	32	32.475	83.3%	32.433	84.7%	34.847
rand_n100_p010	100	489	32	32.029	98.5%	32.151	92.5%	34.020
rand_n200_p002	200	407	95	95.778	0.0%	95.043	94.5%	95.778
rand_n200_p003	200	631	81	82.425	46.4%	81.079	97.0%	83.662
rand_n200_p004	200	816	67	69.890	58.1%	69.818	59.2%	73.908
rand_n200_p005	200	991	64	67.355	33.4%	65.544	69.3%	69.039
reg_n100_r10	100	474	28	29.636	56.9%	29.431	62.3%	31.797
reg_n100_r4	100	195	40	40.333	90.3%	40.713	79.3%	43.449
reg_n100_r6	100	294	34	34.667	82.5%	35.047	72.5%	37.815
reg_n100_r8	100	377	31	31.645	81.4%	32.063	69.4%	34.480
reg_n200_r10	200	980	[57, 59]	60.052	67.5%	62.695	39.5%	66.418
reg_n200_r4	200	400	81	82.450	78.5%	82.246	81.5%	87.759
reg_n200_r6	200	593	[69, 72]	72.685	64.1%	73.709	54.1%	79.276
reg_n200_r8	200	792	[60, 63]	64.749	55.9%	66.789	37.0%	70.790

Table 4: Results on evil, random and near-regular graphs.

numbers of the DIMACS graphs are known, see e.g., [4].

For all graphs, we use the method from Section 4.1 to select a basis of size at most 2500 for the ADMM. The tested graphs on $n \leq 80$ vertices satisfy $\left| \binom{[n]}{\leq 2} \cap [n]_G \right| \leq 2500$, which implies that we compute the full second level of the Lasserre hierarchy for those graphs. The results are reported in Table 7, with the same column definitions as in Table 4, except for the new column denoted by $|\mathcal{B}|$. This column reports the size of the used basis (in Table 4, $|\mathcal{B}| = 2500$ for every graph). Bounds that equal $\alpha(G)$ when rounded down are boldfaced.

We again compare our SDP-Lasserre bounds with BOUND 2 from [34]. Both these bounds are stronger than the bounds in [12]. Computing BOUND 2 for the graphs in Table 7 requires on average only two minutes of computation time per graph. However, for the larger graphs ($n \geq 150$), BOUND 2 cannot be computed due to insufficient memory, as indicated by the table entry ‘-’. The large memory requirement of BOUND 2 is due to the large number of linear constraints. The SDP-Lasserre bounds improve significantly over $\vartheta(G)$, and often also over BOUND 2. In particular, the SDP-Lasserre bounds often close the gap towards $\alpha(G)$ when rounded down. All graphs with $n < 72$ took at most 122 seconds, except for G_60_025 which required approximately 270 seconds. For graphs with $n \geq 72$, the ADMM

Graph name	n	$ E $	$\alpha(G)$ bounds		SDP-Lasserre	$ \mathcal{B} $	T. (min.)
			New	Old [14]			
reg_n200_r10	200	980	[57, 58]	[57, 59]	58.932	4250	180
reg_n200_r6	200	593	[69, 71]	[69, 72]	71.821	3750	105

Table 5: Improved bounds on $\alpha(G)$ with larger basis size $|\mathcal{B}|$ and longer running time.

Graph name	$\alpha(G)$	SDP-Lasserre		BOUND 2 [34]		ESH [13]		$\vartheta(G)$
		Bound	GapClsd	Bound	GapClsd	Bound	GapClsd	
rand_n100_p004	45	45.140	86.8%	45.027	97.4%	45.021	98.0%	46.067
rand_n100_p006	38	38.199	91.5%	38.435	81.5%	38.439	81.4%	40.361
rand_n100_p008	32	32.475	83.3%	32.433	84.7%	32.579	79.6%	34.847
rand_n100_p010	32	32.029	98.5%	32.151	92.5%	32.191	90.5%	34.020
rand_n200_p002	95	95.778	0.0%	95.043	94.5%	95.032	95.8%	95.778
rand_n200_p003	81	82.425	46.4%	81.079	97.0%	81.224	91.5%	83.662
rand_n200_p004	67	69.890	58.1%	69.818	59.2%	70.839	44.4%	73.908
rand_n200_p005	64	67.355	33.4%	65.544	69.3%	66.091	58.5%	69.039
reg_n100_r4	40	40.333	90.3%	40.713	79.3%	40.687	80.0%	43.449
reg_n100_r6	34	34.667	82.5%	35.047	72.5%	35.246	67.3%	37.815
reg_n100_r8	31	31.645	81.4%	32.063	69.4%	32.190	65.8%	34.480
reg_n200_r10	[57, 59]	60.052	67.5%	62.695	39.5%	62.894	37.4%	66.418
reg_n200_r4	81	82.450	78.5%	82.246	81.5%	83.732	59.5%	87.759
reg_n200_r6	[69, 72]	72.685	64.1%	73.709	54.1%	75.555	36.2%	79.276
reg_n200_r8	[60, 63]	64.749	55.9%	66.789	37.0%	67.785	27.8%	70.790

Table 6: Comparison of the SDP-Lasserre, BOUND 2 and ESH bounds.

algorithm required between 15 and 60 minutes to terminate.

For the graph `c_fat200_5` in Table 7, the SDP-Lasserre bound does not improve over $\vartheta(G) = 60.345$. We ran our ADMM algorithm for 4 hours to recompute the SDP-Lasserre bound of this graph, which resulted in an improved bound of 60.317, see also Table 8.

5.3.3 SDP bounds on SageMath graphs

We compute the SDP-Lasserre bounds for several graphs from the SageMath [42] software. Specifically, we consider SageMath graphs on at least 30 vertices, for which $\vartheta(G)$ is strictly larger than $\alpha(G)$.

The results are reported in Table 9, which uses the same column definitions as Table 7. The SDP-Lasserre bounds improve significantly over $\vartheta(G)$, and all of them equal $\alpha(G)$ when rounded down. On average, the ADMM algorithm required 1098 seconds to terminate for each graph. It is worth noting that for each graph, the ADMM required at most 900 seconds to reach an iteration ℓ for which $\lfloor v(X^\ell) \rfloor = \alpha(G)$, see (3). The computation of BOUND 2 required at most 220 seconds, but there are some graphs in Table 9 for which the floor of BOUND 2 does not equal $\alpha(G)$.

6 Conclusions and future work

We have considered SDP bounds on the stability number of graphs obtained from the Lasserre hierarchy at relaxation level 2, or relaxation levels intermediate to levels 1 and 2. Most of the SDPs considered here cannot be handled by IPMs, as they require excessive memory due to the large number of constraints. Therefore, we compute the bounds using the ADMM. The main operations of the ADMM algorithm are the projection onto the PSD cone (3.2), and the projection onto half-spaces, see Lemma 2. Although the former projection is significantly more computationally expensive than the latter, it is still manageable for the problem sizes considered in this work. For improved performance, our ADMM algorithm also uses warm-starting.

For Lasserre levels intermediate to levels 1 and 2, we propose a method to select a basis of variables for the relaxation, see Section 4.1. We use this basis selection method to choose bases of size at most 2500 for computing bounds on $\alpha(G)$ for various graphs in Section 5.3. With this basis size, the ADMM algorithm often converges within one hour of computation time.

Graph name	n	$ E $	$\alpha(G)$	SDP-Lasserre			BOUND 2 [34]		$\vartheta(G)$
				$ \mathcal{B} $	Bound	GapClsd	Bound	GapClsd	
HoG_34272	9	17	3	29	3.000	99.8%	3.000	99.9%	3.338
HoG_15599	20	44	7	167	7.003	99.6%	7.000	99.9%	7.820
CubicVT26_5	26	39	10	313	10.100	94.5%	10.662	63.5%	11.817
HoG_34274	36	72	12	595	12.010	99.1%	12.000	99.9%	13.232
HoG_6575	45	225	10	811	10.132	97.3%	13.220	36.2%	15.053
MANN_a9_clq	45	72	16	964	16.281	80.9%	17.225	16.9%	17.475
Circulant47_030	47	282	13	847	13.003	99.7%	13.026	97.9%	14.302
G_50_025	50	308	12	968	12.028	98.2%	12.367	76.5%	13.564
G_60_025	60	450	13	1381	13.003	99.7%	13.241	81.1%	14.281
PaleyGraph61	61	915	5	977	5.289	89.7%	7.810	0.0%	7.810
hamming6_4	64	1312	4	769	4.032	97.5%	4.749	43.7%	5.333
HoG_34276	72	144	24	2485	24.042	98.2%	24.000	99.9%	26.463
G_80_050	80	1620	9	1621	9.001	99.7%	9.092	78.8%	9.435
G_100_025	100	1243	17	2500	17.000	99.9%	18.428	41.5%	19.441
spin5	125	375	50	2500	50.236	95.9%	50.000	99.9%	55.902
G_150_025	150	2835	19	2500	19.127	97.3%	—	—	23.718
keller4	171	5100	11	2500	11.622	79.3%	—	—	14.012
G_200_025	200	4905	21	2500	23.656	63.1%	—	—	28.217
brock200_1	200	5066	21	2500	22.912	70.3%	—	—	27.457
c_fat200_5	200	11427	58	2500	60.345	0.0%	—	—	60.345
sanr200_0_9	200	2037	42	2500	43.856	74.4%	—	—	49.274

Table 7: Results on the graphs from [12].

Graph name	n	$ E $	$\alpha(G)$	SDP-Lasserre bound after		$\vartheta(G)$
				1 hour	4 hours	
rand_n200_p002	200	407	95	95.778	95.244	95.778
c_fat200_5	200	11427	58	60.345	60.317	60.345

Table 8: Improved SDP-Lasserre bounds with longer ADMM running times.

The computational experiments in Section 5.3 show that the Lasserre hierarchy bounds, computed via the ADMM and referred to as SDP-Lasserre bounds, are competitive with other SDP-based stable set approaches from the literature, specifically [13] and [34]. In particular, our approach provides the strongest known SDP bounds on $\alpha(G)$ for a variety of graphs, see Tables 4 to 9. For some large and dense graphs in Table 7, the bound from [34] cannot be computed using the IPM due to its large memory requirement, in contrast to our SDP-Lasserre bounds.

As future work, it would be interesting to evaluate bounds from the intermediate level Lasserre hierarchy on the stability number of highly symmetric graphs. Specifically, our basis selection method from Section 4.1 is currently not suited to exploit symmetry in the underlying graph. Preliminary computational experiments have shown that the current basis selection method can be significantly improved for highly symmetric graphs. Another future research direction is to investigate faster methods for projecting onto the PSD cone, which is the bottleneck of the ADMM. If we find an algorithm for projecting onto the PSD cone that is better suited for the ADMM than the standard eigenvalue decomposition, the ADMM could speed up significantly.

Statements and Declarations

Conflict of interest. The authors declare that they have no conflict of interest.

Funding. No funds, grants, or other support was received.

Availability of data and materials. All the graphs used in this work are available online, or on request.

Ethics approval and consent to participate. Not applicable.

Graph name	n	$ E $	$\alpha(G)$	SDP-Lasserre			BOUND 2 [34]		$\vartheta(G)$
				$ \mathcal{B} $	Bound	GapClsd	Bound	GapClsd	
DoubleStarSnark	30	45	13	421	13.001	99.9%	13.001	99.8%	13.735
Wells	32	80	10	449	10.001	99.9%	10.906	54.7%	12.000
Sylvester	36	90	12	577	12.001	99.9%	12.034	97.7%	13.500
SzekeresSnark	50	75	21	1201	21.035	97.7%	21.359	76.6%	22.537
Klein3RegularGraph	56	84	23	1513	23.059	97.8%	23.908	67.4%	25.793
Gosset	56	756	4	841	4.048	97.0%	5.600	0.0%	5.600
Gritsenko	65	1040	6	1106	6.097	95.2%	8.062	0.0%	8.062
Meredith	70	140	34	2346	34.094	80.7%	34.003	99.3%	34.489
BrouwerHaemers	81	810	15	2500	15.041	99.3%	20.259	12.3%	21.000
HigmanSimsGraph	100	1100	22	2500	22.005	99.8%	26.667	0.0%	26.667
BiggsSmith	102	153	43	2500	43.536	85.4%	44.270	65.5%	46.686
Balaban11Cage	112	168	52	2500	52.124	92.2%	52.064	95.9%	53.595

Table 9: Results on several graphs from the SageMath software.

Consent for publication. Not applicable.

Acknowledgements. Not applicable.

References

- [1] E. Adams, M. F. Anjos, F. Rendl, and A. Wiegele. A hierarchy of subgraph projection-based semidefinite relaxations for some NP-hard graph optimization problems. *INFOR: Information Systems and Operational Research*, 53(1):40–48, 2015.
- [2] F. Alizadeh. *Combinatorial optimization with interior point methods and semidefinite matrices*. PhD thesis, University of Minnesota, Minneapolis, USA, 1991.
- [3] M. F. Anjos. An improved semidefinite programming relaxation for the satisfiability problem. *Mathematical Programming*, 102:589–608, 2005.
- [4] E. Balas and J. Xue. Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring. *Algorithmica*, 15(5):397–412, 1996.
- [5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.
- [6] G. Brinkmann, K. Coolsaet, J. Goedgebeur, and H. Mélot. House of graphs: a database of interesting graphs. *Discrete Applied Mathematics*, 161(1-2):311–314, 2013.
- [7] J. S. Campos, R. Misener, and P. Parpas. Partial Lasserre relaxation for sparse max-cut. *Optimization and Engineering*, 24:1983–2004, 2023.
- [8] T. Chen, J.-B. Lasserre, V. Magron, and E. Pauwels. A sublevel moment-SOS hierarchy for polynomial optimization. *Computational Optimization and Applications*, 81(1):31–66, 2022.
- [9] E. de Klerk. Exploiting special structure in semidefinite programming: A survey of theory and applications. *European Journal of Operational Research*, 201:1–10, 2010.
- [10] I. Dukanovic and F. Rendl. Semidefinite programming relaxations for graph coloring and maximal clique problems. *Mathematical Programming*, 109(2):345–365, 2007.
- [11] M. Fortin and R. Glowinski. On decomposition-coordination methods using an augmented Lagrangian. In M. Fortin and R. Glowinski, editors, *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, volume 15, pages 97–146. Elsevier, 1983.
- [12] E. Gaar. On different versions of the exact subgraph hierarchy for the stable set problem. *Discrete Applied Mathematics*, 356:52–70, 2024.

- [13] E. Gaar and F. Rendl. A computational study of exact subgraph based SDP bounds for max-cut, stable set and coloring. *Mathematical Programming*, 183(1):283–308, 2020.
- [14] E. Gaar, M. Siebenhofer, and A. Wiegele. An SDP-based approach for computing the stability number of a graph. *Mathematical Methods of Operations Research*, 95(1):141–161, 2022.
- [15] J. Gouveia, P. A. Parrilo, and R. R. Thomas. Theta bodies for polynomial ideals. *SIAM Journal on Optimization*, 20(4):2097–2118, 2010.
- [16] G. Gruber and F. Rendl. Computational experience with stable set relaxations. *SIAM Journal on Optimization*, 13(4):1014–1028, 2003.
- [17] B. He, F. Ma, and X. Yuan. Convergence study on the symmetric version of ADMM with larger step sizes. *SIAM Journal on Imaging Sciences*, 9(3):1467–1501, 2016.
- [18] I. D. Ivanov and E. de Klerk and. Parallel implementation of a semidefinite programming solver based on CSDP on a distributed memory cluster. *Optimization Methods and Software*, 25(3):405–420, 2010.
- [19] W. Karush. Minima of functions of several variables with inequalities as side constraints. *MSc. Thesis. Department of Mathematics, University of Chicago, Chicago, Illinois*, 1939.
- [20] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492. Springer, 1951.
- [21] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.
- [22] J. B. Lasserre. *Moments, positive polynomials and their applications*, volume 1. World Scientific, 2009.
- [23] M. Laurent. A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0–1 programming. *Mathematics of Operations Research*, 28(3):470–496, 2003.
- [24] M. Laurent. Semidefinite representations for finite varieties. *Mathematical Programming*, 109(1):1–26, 2007.
- [25] M. Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging Applications of Algebraic Geometry*, pages 157–270. Springer, 2009.
- [26] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information theory*, 25(1):1–7, 1979.
- [27] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0–1 optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.
- [28] R. Madani, A. Kalbat, and J. Lavaei. ADMM for sparse semidefinite programming with applications to optimal power flow problem. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 5932–5939. IEEE, 2015.
- [29] R. D. C. Monteiro and P. Zanjácomo. Implementation of primal-dual methods for semidefinite programming based on Monteiro and Tsuchiya newton directions and their variants. *Optimization Methods and Software*, 11(1-4):91–140, 1999.
- [30] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 10.1.*, 2024. URL <http://docs.mosek.com/latest/toolbox/index.html>.
- [31] Y. Nesterov and A. Nemirovskii. Interior point polynomial algorithms in convex programming. In *SIAM Studies in Applied Mathematics*. SIAM, Philadelphia, USA, Vol. 13, 1994.
- [32] D. E. Oliveira, H. Wolkowicz, and Y. Xu. ADMM for the SDP relaxation of the QAP. *Mathematical Programming Computation*, 10(4):631–658, 2018.
- [33] P. A. Parrilo. An explicit construction of distinguished representations of polynomials nonnegative over finite sets. *Preprint, ETH, Zürich*, 2002.

- [34] D. Pucher and F. Rendl. Practical experience with stable set and coloring relaxations. *arXiv preprint arXiv:2401.17069v2*, 2024.
- [35] P. Raghavendra and B. Weitz. On the bit complexity of sum-of-squares proofs. In I. Chatzigiannakis, P. Indyk, F. Kuhn, and A. Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming*, volume 80 of *LIPICs*, pages 80:1–80:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.
- [36] N. Rontsis, P. Goulart, and Y. Nakatsukasa. Efficient semidefinite programming with approximate ADMM. *Journal of Optimization Theory and Applications*, 192:292–320, 2022.
- [37] A. Schrijver. A comparison of the Delsarte and Lovász bounds. *IEEE Transactions on Information Theory*, 25(4):425–429, 1979.
- [38] H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3(3):411–430, 1990.
- [39] F. Silvestri. Spectrahedral relaxations of the stable set polytope using induced subgraphs. Master’s thesis, Universität Heidelberg, 2013.
- [40] L. Sinjorgo and R. Sotirov. On solving MAX-SAT using sum of squares. *INFORMS Journal on Computing*, 36(2):417–433, 2024.
- [41] S. Szabó and B. Zaválnij. Benchmark problems for exhaustive exact maximum clique search algorithms. *Informatica*, 43(2):177–186, 2019.
- [42] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 10.5)*, 2025. <https://www.sagemath.org>.
- [43] H. van Maaren, L. van Norden, and M. J. Heule. Sums of squares based approximation algorithms for MAX-SAT. *Discrete Applied Mathematics*, 156(10):1754–1779, 2008.