Numerical block diagonalization of matrix $*$-algebras with application to semidefinite programming

Etienne de Klerk[*] Cristian Dobre[†] Dmitrii V. Pasechnik[‡]

November 21, 2009

Abstract

Semidefinite programming (SDP) is one of the most active areas in mathematical programming, due to varied applications and the availability of interior point algorithms. In this paper we propose a new pre-processing technique for SDP instances that exhibit algebraic symmetry. We present computational results to show that the solution times of certain SDP instances may be greatly reduced via the new approach.

Keywords: semidefinite programming, algebraic symmetry, pre-processing, interior point methods

AMS classification: 90C22

[*]Department of Econometrics and OR, Tilburg University, The Netherlands. e.deklerk@uvt.nl

[†]Department of Econometrics and OR, Tilburg University, The Netherlands. c.dobre@uvt.nl

[‡]Division of Mathematical Sciences, SPMS, Nanyang Technological University, 21 Nanyang Link, 637371 Singapore. dima@ntu.edu.sg

1 Introduction

Semidefinite programming (SDP) is currently one of the most active areas of research in mathematical programming. The reason for this is two-fold: applications of SDP may be found in control theory, combinatorics, real algebraic geometry, global optimization and structural design, to name only a few; see the surveys by Vandenberghe and Boyd [29] and Todd [26] for more information. The second reason is the extension of interior point methods from linear programming (LP) to SDP in the 1990's by Nesterov and Nemirovski [22], Alizadeh [1], and others.

A recurrent difficulty in applying interior point methods for SDP is that it is more difficult to exploit special structure in the data than in the LP case. In particular, sparsity may readily be exploited by interior point methods in LP, but this is not true for SDP. Some structures in SDP data have been exploited successfully, see the recent survey [11] for details.

Of particular interest for this paper is a structure called algebraic symmetry, where the SDP data matrices are contained in a low-dimensional matrix $\mathbb{C}*$-algebra. (Recall that a matrix $*$-algebra is a linear subspace of $\mathbb{C}^{n\times n}$ that is closed under multiplication and taking complex conjugate transposes.) Although this structure may seem exotic, it arises in a surprising number of applications, and first appeared in a paper by Schrijver [23] in 1979 on bounds for binary code sizes. (Another early work on algebraic symmetry in SDP is by Kojima *et al.* [16].)

More recent applications are surveyed in [11, 7, 28] and include bounds on kissing numbers [2], bounds on crossing numbers in graphs [12, 13], bounds on code sizes [24, 9, 17], truss topology design [10, 3], quadratic assignment problems [14], etc.

Algebraic symmetry may be exploited since matrix $\mathbb{C}*$-algebras have a canonical block diagonal structure after a suitable unitary transform. Block diagonal structure may in turn be exploited by interior point algorithms.

For some examples of SDP instances with algebraic symmetry, the required unitary transform is known beforehand, *e.g.* as in [24]. For other examples, like the instances in [12, 13], it is not.

In cases like the latter, one may perform numerical pre-processing in order to obtain the required unitary transformation. A suitable algorithm is given in [5], but the focus there is on complexity and symbolic computation, as opposed to practical floating point computation. Murota *et al.* [21] presented a practical randomized algorithm that may be used for pre-processing of SDP instances with algebraic symmetry; this work has recently been extended by Maehara and Murota [19].

In this paper, we propose another numerical pre-processing approach in the spirit of the work by Murota *et al.* [21], although the details are somewhat different. We demonstrate that the new approach may offer numerical advantages for certain group symmetric SDP instances, in particular for the SDP instances from [13]. In particular, we show how to solve one specific instance from [13] in a few minutes on a PC after pre-processing, where the original solution (reported in [13]) required a week on a supercomputer.

Outline

This paper is structured as follows. After a summary of notation we review the basic properties of matrix $\mathbb{C}*$-algebras in Section 2. In particular, the canonical block decomposition is described there, and two conceptual algorithms to compute it in two steps are given in Sections 3 and 4.

In this paper we will apply these algorithms not to the given matrix $\mathbb{C}*$-algebra, but to its so-called regular $*$-representation, described in Section 5.

In Section 6 we review how these algorithms may be used to reduce the size of SDP instances with algebraic symmetry, and we summarize the numerical algorithm that we propose in this paper.

We relate our approach to the earlier work of Murota *et al.* [21] in Section 7, and conclude with Section 8 on numerical experiments.

Notation and preliminaries

The space of $p \times q$ real (resp. complex) matrices will be denoted by $\mathbb{R}^{p\times q}$ (resp. $\mathbb{C}^{p\times q}$), and the space of $k \times k$ symmetric matrices by $\mathbb{S}^{k\times k}$.

We use I_n to denote the identity matrix of order n. Similarly, J_n denotes the $n \times n$ all-ones matrix. We will omit the subscript if the order is clear from the context.

A complex matrix $A \in \mathbb{C}^{n\times n}$ may be decomposed as

$$A = \mathrm{Re}(A) + \sqrt{-1}\mathrm{Im}(A),$$

where $\mathrm{Re}(A) \in \mathbb{R}^{n\times n}$ and $\mathrm{Im}(A) \in \mathbb{R}^{n\times n}$ are the real and imaginary parts of A, respectively. The *complex conjugate transpose* is defined as:

$$A^* = \mathrm{Re}(A)^T - \sqrt{-1}\mathrm{Im}(A)^T,$$

where the superscript T denotes the transpose.

A matrix $A \in \mathbb{C}^{n\times n}$ is called *Hermitian* if $A^* = A$, i.e. if $\mathrm{Re}(A)$ is symmetric and $\mathrm{Im}(A)$ is skew-symmetric. If A is Hermitian, then $A \succeq 0$ means A is *positive semidefinite*. A matrix $Q \in \mathbb{C}^{n\times n}$ is called *unitary* if $Q^*Q = I$. A real unitary matrix is called *orthogonal*.

2 Basic properties of matrix *-algebras

In what follows we give a review of decompositions of matrix $*$-algebras over $\mathbb{C}$, with an emphasis on the constructive (algorithmic) aspects. Our exposition and notation is based on the PhD thesis of Gijswijt [8], §2.2.

Definition 1. *A set $\mathcal{A} \subseteq \mathbb{C}^{n\times n}$ is called a matrix *-algebra over $\mathbb{C}$ (or a matrix $\mathbb{C}*$-algebra) if, for all $X, Y \in \mathcal{A}$:*

- $\alpha X + \beta Y \in \mathcal{A} \quad \forall \alpha, \beta \in \mathbb{C}$*;*

- *$X^* \in \mathcal{A}$;*
- *$XY \in \mathcal{A}$.*

A matrix $\mathbb{C}*$-subalgebra of $\mathcal{A}$ is called maximal, if it is not properly contained in any other proper $\mathbb{C}*$-subalgebra of $\mathcal{A}$.

In applications one often encounters matrix $\mathbb{C}*$-algebras with the following additional structure.

Definition 2. *A matrix $\mathbb{C}*$-algebra is called a coherent configuration, if it contains the identity matrix and has a basis of zero-one matrices that sum to the all ones matrix.*

More information on coherent configurations and related structures may be found in [4].

The *direct sum* of two square matrices A_1, A_2, is defined as

$$A_1 \oplus A_2 := \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}, \tag{1}$$

and the iterated direct sum of square matrices $A_1, ..., A_n$ is denoted by $\bigoplus_{i=1}^n A_i$. If all the A_i matrices are equal we define:

$$t \odot A := \bigoplus_{i=1}^{t} A.$$

Let $\mathcal{A}$ and $\mathcal{B}$ be two matrix $\mathbb{C}*$-algebras. Then the direct sum of the two algebras $\mathcal{A}$ and $\mathcal{B}$ is defined as:

$$\mathcal{A} \oplus \mathcal{B} := \{M \oplus M' \mid M \in \mathcal{A}, M' \in \mathcal{B}\}.$$

Note that the "$\oplus$" symbol is used to denote both the direct sum of matrices and the direct sum of algebras. Since these are conceptually completely different operations, we will always denote algebras with calligraphic capitals, like $\mathcal{A}$, and matrices using capitals in normal font, like A, to avoid confusion.

We say that $\mathcal{A}$ is a zero algebra if it consists only of the zero matrix.

Definition 3. *A matrix $\mathbb{C}$*-algebra $\mathcal{A}$ is called basic if*

$$\mathcal{A} = t \odot \mathbb{C}^{s\times s} := \{t \odot M \mid M \in \mathbb{C}^{s\times s}\} \tag{2}$$

for some integers s, t.

Note that each eigenvalue of a generic element of the basic $\mathbb{C}*$-algebra in (2) has multiplicity t.

Definition 4. *Two matrix $\mathbb{C}*$-algebras $\mathcal{A}, \mathcal{B} \subset \mathbb{C}^{n\times n}$ are called equivalent if there exists a unitary matrix $Q \in \mathbb{C}^{n\times n}$ such that*

$$\mathcal{B} = \{Q^* M Q \mid M \in \mathcal{A}\} =: Q^* \mathcal{A} Q.$$

It is well known that every simple matrix $\mathbb{C}*$-algebra that contains the identity is equivalent to a basic algebra. (Recall that a matrix $\mathbb{C}*$-algebra is called simple if it has no nontrivial ideal.)

The fundamental decomposition theorem for matrix $\mathbb{C}*$-algebra states the following.

Theorem 1 (Wedderburn (1907) [30])**.** *Each matrix *-algebra over $\mathbb{C}$ is equivalent to a direct sum of basic algebras and possibly a zero algebra.*

A detailed proof of this result is given *e.g.* in the thesis by Gijswijt [8] (Proposition 4 there). The proof is constructive, and forms the basis for numerical procedures to obtain the decomposition into basic algebras.

3 Constructing the Wedderburn decomposition

Let center($\mathcal{A}$) denote the center of a given matrix $\mathbb{C}*$-algebra $\mathcal{A} \subset \mathbb{C}^{n\times n}$:

$$\text{center}(\mathcal{A}) = \{X \in \mathcal{A} \mid XA = AX \text{ for all } A \in \mathcal{A}\}.$$

The center is a commutative sub-algebra of $\mathcal{A}$ and therefore has a common set of orthonormal eigenvectors that we may view as the columns of a unitary matrix Q, i.e. $Q^*Q = I$. We arrange the columns of Q such that eigenvectors corresponding to one eigenspace are grouped together. The center also has a basis of idempotents (see *e.g.* Proposition 1 in [8]), say $E_1, \ldots, E_t$. If $\mathcal{A}$ contains the identity I, then $\sum_{i=1}^t E_i = I$. In what follows we assume that $\mathcal{A}$ contains the identity.

The unitary transform $\mathcal{A}' := Q^*\mathcal{A}Q$ transforms the E_i matrices to zero-one diagonal matrices (say $E_i' := Q^*E_iQ$) that sum to the identity.

One clearly has

$$\mathcal{A}' = \sum_{i=1}^t \mathcal{A}'E_i' = \sum_{i=1}^t \mathcal{A}'E_i'^2 = \sum_{i=1}^t E_i'\mathcal{A}'E_i'.$$

Each term $E_i'\mathcal{A}'E_i'$ $(i = 1, \ldots, t)$ is clearly a matrix $\mathbb{C}*$-algebra. For a fixed i, the matrices in the algebra $E_i'\mathcal{A}'E_i'$ have a common nonzero diagonal block indexed by the positions of the ones on the diagonal of E_i'. Define $\mathcal{A}_i$ as the restriction of $E_i'\mathcal{A}'E_i'$ to this principal submatrix. One now has

$$Q^*\mathcal{A}Q = \oplus_{i=1}^t \mathcal{A}_i. \tag{3}$$

Moreover, one may show (see Proposition 4 in [8]), that each $\mathcal{A}_i$ is a simple algebra, i.e. it has no nontrivial ideal.

Numerically, the decomposition (3) of $\mathcal{A}$ into simple components may be done using the following framework algorithm.

In step (i), we assume that a basis of center($\mathcal{A}$) is available. The generic element X is then obtained by taking a random linear combination of the basis elements. This approach is also used by Murota *et al.* [21], see *e.g.* Algorithm 4.1. there.

Algorithm 1 Decomposition of $\mathcal{A}$ into simple components

```
**INPUT:** A $\mathbb{C}*$-algebra $\mathcal{A}$.
(i) Sample a generic element, say $X$, from center($\mathcal{A}$);
(ii) Perform the spectral decomposition of $X$ to obtain a unitary matrix $Q$ containing a set of orthonormal eigenvectors of $X$;
**OUTPUT:** a unitary matrix $Q$ such that $Q^*\mathcal{A}Q$ gives the decomposition (3).
```

4 From simple to basic components

Recall that all simple matrix $*$-algebras over $\mathbb{C}$ are equivalent to basic algebras. Thus we may still decompose each $\mathcal{A}_i$ in (3) as

$$U_i^*\mathcal{A}_iU_i = t_i \odot \mathbb{C}^{n_i\times n_i}$$

for some integers n_i and t_i and some unitary matrix U_i $(i = 1,\ldots,t)$. For the dimensions to agree, one must have

$$\sum_{i=1}^{t} n_it_i = n \text{ and } \dim(\mathcal{A}) = \sum_{i=1}^{t} n_i^2,$$

since $Q^*\mathcal{A}Q = \oplus_{i=1}^{t}\mathcal{A}_i \subset \mathbb{C}^{n\times n}$.

Now let $\mathcal{B}$ denote a given basic matrix $*$-algebra over $\mathbb{C}$. One may compute the decomposition $U^*\mathcal{B}U = t \odot \mathbb{C}^{s\times s}$, say, where U is unitary and s and t are integers, as follows.

Algorithm 2 Decomposition of a basic $\mathcal{B}$ into basic components

```
**INPUT:** A basic $\mathbb{C}*$-algebra $\mathcal{B}$.
(i) Sample a generic element, say $B$, from any maximal commutative matrix $\mathbb{C}*$-sub-algebra of $\mathcal{B}$.
(ii) Perform a spectral decomposition of $B$, and let $Q$ denote the unitary matrix of its eigenvectors.
(iii) Partition $Q^*\mathcal{B}Q$ into $t\times t$ square blocks, each of size $s\times s$, where $s$ is the number of distinct eigenvalues of $B$.
(iv) Sample a generic element from $Q^*\mathcal{B}Q$, say $B'$, and denote the $ij$th block by $B'_{ij}$. We may assume that $B'_{11},\ldots,B'_{1t}$ are unitary matrices (possibly after a suitable constant scaling).
(v) Define the unitary matrix $Q' := \oplus_{i=1}^{t}(B'_{1i})^*$ and replace $Q^*\mathcal{B}Q$ by $Q'^*Q^*\mathcal{B}QQ'$. Each block in the latter algebra equals $\mathbb{C}I_s$.
(vi) Permute rows and columns to obtain $P^TQ'^*Q^*\mathcal{B}QQ'P = t\odot\mathbb{C}^{s\times s}$, where $P$ is a suitable permutation matrix.
**OUTPUT:** a unitary matrix $U := QQ'P$ such that $U^*\mathcal{B}U = t\odot\mathbb{C}^{s\times s}$.
```

A few remarks on Algorithm 2:

- Step (i) in Algorithm 2 may be performed by randomly sampling a generic element from $\mathcal{B}$.
- By the proof of Proposition 5 in [8], the diagonal blocks in step (iii) are the algebras $\mathbb{C}I_s$.
- By the proof of Proposition 5 in [8], the blocks $B'_{11}, \ldots, B'_{1t}$ used in step (v) are all unitary matrices (up to a constant scaling), so that Q' in step (v) is unitary too.

5 The regular *-representation of $\mathcal{A}$

In this paper we will not compute the Wedderburn decomposition of a given $\mathbb{C}*$-algebra $\mathcal{A}$ directly, but will compute the Wedderburn decomposition of a faithful (i.e. isomorphic) representation of it, called the regular $*$-representation of $\mathcal{A}$. This allows numerical computation with smaller matrices. Moreover, we will show that it is relatively simple to obtain a generic element from the center of the regular $*$-representation of $\mathcal{A}$.

Assume now that $\mathcal{A}$ has an orthogonal basis of real matrices $B_1, \ldots, B_d \in \mathbb{R}^{n \times n}$. This situation is not generic, but it is usual for the applications in semidefinite programming that we will consider.

We normalize this basis with respect to the *Frobenius norm*:

$$D_i := \frac{1}{\sqrt{\operatorname{trace}(B_i^T B_i)}} B_i \quad (i = 1, \ldots, d),$$

and define multiplication parameters $\gamma_{i,j}^k$ via:

$$D_i D_j = \sum_k \gamma_{i,j}^k D_k,$$

and subsequently define the $d \times d$ matrices L_i $(i = 1, \ldots, d)$ via

$$(L_i)_{k,j} := \gamma_{i,j}^k \qquad (i, j, k = 1, \ldots, d).$$

The matrices L_k form a basis of a faithful (i.e. isomorphic) representation of $\mathcal{A}$, say $\mathcal{A}^{reg}$, that is also a matrix $*$-algebra, called the *regular $*$-representation of $\mathcal{A}$.*

Theorem 2 (cf. [13]). *The bijective linear mapping $\phi : \mathcal{A} \mapsto \mathcal{A}^{reg}$ such that $\phi(D_i) = L_i$ $(i = 1, \ldots, d)$ defines a $*$-isomorphism from $\mathcal{A}$ to $\mathcal{A}^{reg}$. Thus, ϕ is an algebra isomorphism with the additional property*

$$\phi(A^*) = \phi(A)^* \quad \forall A \in \mathcal{A}.$$

Since ϕ is a homomorphism, A and $\phi(A)$ have the same eigenvalues (up to multiplicities) for all $A \in \mathcal{A}$. As a consequence, one has

$$\sum_{i=1}^d x_i D_i \succeq 0 \Longleftrightarrow \sum_{i=1}^d x_i L_i \succeq 0.$$

We note that the proof in [13] was only stated for the case that $\mathcal{A}$ is the commutant of a group of permutation matrices, but the proof given there remains valid for any matrix $\mathbb{C}*$-algebra.

By the Wedderburn theorem, any matrix $\mathbb{C}*$-algebra $\mathcal{A}$ takes the form

$$Q^*\mathcal{A}Q = \oplus_{i=1}^{t} t_i \odot \mathbb{C}^{n_i \times n_i}, \tag{4}$$

for some integers t, t_i and n_i $(i = 1, \ldots, t)$, and some unitary Q.

It is easy to verify that the regular $*$-representations of $\mathcal{A}$ and $Q^*\mathcal{A}Q$ are the same. This implies that, when studying $\mathcal{A}^{reg}$, we may assume without loss of generality that $\mathcal{A}$ takes the form

$$\mathcal{A} = \oplus_{i=1}^{t} t_i \odot \mathbb{C}^{n_i \times n_i}.$$

Our goal here is to show that the Wedderburn decomposition of $\mathcal{A}^{reg}$ has a special structure that does not depend on the values t_i $(i = 1, \ldots, t)$.

To this end, the basic observation that is needed is given in the following lemma. The proof is straightforward, and therefore omitted.

Lemma 1. *Let t and n be given integers. The regular *-representation of $t \odot \mathbb{C}^{n\times n}$ is equivalent to $n \odot \mathbb{C}^{n\times n}$.*

Using the last lemma, one may readily prove the following.

Theorem 3. *The regular *-representation of $\mathcal{A} := \oplus_{i=1}^{t} t_i \odot \mathbb{C}^{n_i \times n_i}$ is equivalent to $\oplus_{i=1}^{t} n_i \odot \mathbb{C}^{n_i \times n_i}$.*

The Wedderburn decomposition of $\mathcal{A}^{reg}$ therefore takes the form

$$U^*\mathcal{A}^{reg}U = \oplus_{i=1}^{t} n_i \odot \mathbb{C}^{n_i \times n_i}, \tag{5}$$

for some suitable unitary matrix U.

Comparing (4) and (5), we may informally say that "the t_i and n_i values are equal for all i" in the Wedderburn decomposition of a regular $*$-representation. We will also observe this in the numerical examples in Section 8.

Sampling from the center of a regular $*$-representation.

In order to compute the Wedderburn decomposition of $\mathcal{A}^{reg}$ we need to sample a generic element from center($\mathcal{A}^{reg}$) (see step (i) in Algorithm 1).

To this end, assume $X := \sum_{k=1}^{d} x_k L_k$ is in the center of $\mathcal{A}^{reg}$. This is the same as assuming for $j = 1, ..., d$:

$$\begin{aligned}
XL_j = L_jX \quad &\Leftrightarrow \quad \sum_{i=1}^{d} x_i L_j L_i = \sum_{i=1}^{d} x_i L_i L_j \\
&\Leftrightarrow \quad \sum_{i=1}^{d} x_i \sum_{k} (L_j)_{ki} L_k = \sum_{i=1}^{d} x_i \sum_{k} (L_i)_{kj} L_k \\
&\Leftrightarrow \quad \sum_{k} \left(\sum_{i=1}^{d} x_i (L_j)_{ki} - \sum_{i=1}^{d} x_i (L_i)_{kj} \right) L_k = 0 \\
&\Leftrightarrow \quad \sum_{i=1}^{d} x_i \left((L_j)_{ki} - (L_i)_{kj} \right) = 0 \quad \forall k = 1, \ldots, d,
\end{aligned}$$

where the last equality follows from the fact that the L_k's form a basis for $\mathcal{A}^{reg}$.

To sample a generic element from center($\mathcal{A}^{reg}$) we may therefore proceed as outlined in Algorithm 3.

Algorithm 3 Obtaining a generic element of center($\mathcal{A}^{reg}$)

```
**INPUT:** A basis $L_1, \ldots, L_d$ of $\mathcal{A}^{reg}$.
(i) Compute a basis of the nullspace of the linear operator $\mathcal{L} : \mathbb{C}^d \to \mathbb{C}^{d^2}$ given by
    $\mathcal{L}(x) = \left[ \sum_{i=1}^{d} x_i \left( (L_j)_{ki} - (L_i)_{kj} \right) \right]_{j,k=1,\ldots,d}$.
(ii) Take a random linear combination of the basis elements of the nullspace of $\mathcal{L}$ to obtain a generic element, say $\bar{x}$, in the nullspace of $\mathcal{L}$.
**OUTPUT:** $\bar{X} := \sum_{k=1}^{d} \bar{x}_k L_k$, a generic element of center($\mathcal{A}^{reg}$).
```

6 Symmetry reduction of SDP instances

We consider the standard form SDP problem

$$\min_{X \succeq 0} \left\{ \mathrm{trace}(A_0 X) \; : \; \mathrm{trace}\left(A_k X\right) = b_k \;\; \forall \; k = 1, \ldots, m \right\}, \tag{6}$$

where the Hermitian data matrices $A_i = A_i^* \in \mathbb{C}^{n \times n}$ $(i = 0, \ldots, m)$ are linearly independent.

We say that the SDP data matrices exhibit *algebraic symmetry* if the following assumption holds.

Assumption 1 (Algebraic symmetry). *There exists a matrix $\mathbb{C}*$-algebra, say $\mathcal{A}_{SDP}$ with dim($\mathcal{A}_{SDP}$) $\ll n$, that contains the data matrices $A_0, \ldots, A_m$.*

Under this assumption, one may restrict the feasible set of problem (6) to its intersection with $\mathcal{A}_{SDP}$, as the following theorem shows. Although results of this type are well-known, we include a proof since we are not aware of the theorem appearing in this form in the literature. Related, but slightly less general results are given in [7, 11] and other papers. The outline of the proof given here was suggested to the authors by Professor Masakazu Kojima.

Theorem 4. *Let $\mathcal{A}_{SDP}$ denote a matrix $\mathbb{C}*$-algebra that contains the data matrices $A_0, \ldots, A_m$ of problem (6) as well as the identity. If problem (6) has an optimal solution, then it has an optimal solution in $\mathcal{A}_{SDP}$.*

Proof. By Theorem 1 we may assume that there exists a unitary matrix Q such that

$$Q^* \mathcal{A}_{SDP} Q = \oplus_{i=1}^{t} t_i \odot \mathbb{C}^{n_i \times n_i}, \tag{7}$$

for some integers t, n_i and t_i $(i = 1, \ldots, t)$.

Since $A_0, \ldots, A_m \in \mathcal{A}$, one has

$$Q^* A_j Q =: \oplus_{i=1}^{t} t_i \odot A_j^{(i)} \quad (j = 0, \ldots, m)$$

for Hermitian matrices $A_j^{(i)} \in \mathbb{C}^{n_i \times n_i}$ where $i = 1, \ldots, t$ and $j = 0, \ldots, m$.

Now assume $\widetilde{X}$ is an optimal solution for (6). We have for each $i = 0, ..., m$:

$$\begin{aligned} \operatorname{trace}(A_j \widetilde{X}) &= \operatorname{trace}(QQ^* A_j QQ^* \widetilde{X}) \\ &= \operatorname{trace}((Q^* A_j Q)(Q^* \widetilde{X} Q)) \\ &= \operatorname{trace} \oplus_{i=1}^{t} t_i \odot A_j^{(i)} Q^* \widetilde{X} Q \\ &=: \operatorname{trace} \oplus_{i=1}^{t} t_i \odot A_j^{(i)} \bar{X}, \end{aligned}$$

where $\bar{X} := Q^* \widetilde{X} Q$.

The only elements of $\bar{X}$ that appear in the last expression are those in the diagonal blocks that correspond to the block structure of $Q^* \mathcal{A} Q$. We may therefore construct a matrix $\bar{X}' \succeq 0$ from $\bar{X}$ by setting those elements of $\bar{X}$ that are outside the blocks to zero, say

$$\bar{X}' = \oplus_{i=1}^{t} \left(\oplus_{k=1}^{t_i} \bar{X}_i^{(k)} \right)$$

where the $\bar{X}_i^{(k)} \in \mathbb{C}^{n_i \times n_i}$ $(k = 1, \ldots, t_i)$ are the diagonal blocks of $\bar{X}$ that correspond to the blocks of the ith basic algebra, i.e. $t_i \odot \mathbb{C}^{n_i \times n_i}$ in (7). Thus we obtain, for $j = 0, \ldots, m$,

$$\begin{aligned}
\operatorname{trace}(A_j\widetilde{X}) &= \operatorname{trace} \oplus_{i=1}^{t} t_i \odot A_j^{(i)} \bar{X}' \\
&= \sum_{i=1}^{t} \sum_{k=1}^{t_i} \operatorname{trace}\left(A_j^{(i)} \bar{X}_i^{(k)}\right) \\
&= \sum_{i=1}^{t} \operatorname{trace}\left(A_j^{(i)} \left[\sum_{k=1}^{t_i} \bar{X}_i^{(k)}\right]\right) \\
&=: \sum_{i=1}^{t} \operatorname{trace}\left((t_i \odot A_j^{(i)})(t_i \odot \bar{X}_i)\right), \qquad (8)
\end{aligned}$$

where $\bar{X}_i = \frac{1}{t_i}\left[\sum_{k=1}^{t_i} \bar{X}_i^{(k)}\right] \succeq 0$. Defining

$$X := \sum_{i=1}^{t} t_i \odot \bar{X}_i$$

one has $X \succeq 0$, $X \in Q^*\mathcal{A}_{SDP}Q$ by (7), and

$$\operatorname{trace}(A_j\widetilde{X}) = \operatorname{trace}(Q^*A_jQX) = \operatorname{trace}(A_jQXQ^*), \quad (j = 0, \ldots, m),$$

by (8). Thus $QXQ^* \in \mathcal{A}_{SDP}$ is an optimal solution of (6). □

In most applications, the data matrices $A_0, \ldots, A_m$ are real, symmetric matrices, and we may assume that $\mathcal{A}_{SDP}$ has a real basis (seen as a subspace of $\mathbb{C}^{n\times n}$). In this case, if (6) has an optimal solution, it has a real optimal solution in $\mathcal{A}_{SDP}$.

Corollary 1. *Assume the data matrices $A_0, \ldots, A_m$ in (6) are real symmetric. If $X \in \mathbb{C}^{n\times n}$ is an optimal solution of problem (6) then Re(X) is also an optimal solution of this problem.*

Moreover, if $\mathcal{A}_{SDP}$ has a real basis, and $X \in \mathcal{A}_{SDP}$, then Re$(X) \in \mathcal{A}_{SDP}$.

Proof. We have $\operatorname{trace}(A_kX) = \operatorname{trace}(A_k\operatorname{Re}(X))$ $(k = 0, \ldots, m)$. Moreover, $X \succeq 0$ implies $\operatorname{Re}(X) \succeq 0$.

The second part of the result follows from the fact that, if $X \in \mathcal{A}_{SDP}$ and $\mathcal{A}_{SDP}$ has a real basis, then both $\operatorname{Re}(X) \in \mathcal{A}_{SDP}$ and $\operatorname{Im}(X) \in \mathcal{A}_{SDP}$. □

By Theorem 4, we may rewrite the SDP problem (6) as:

$$\min_{X\succeq 0} \left\{\operatorname{trace}(A_0X) \;:\; \operatorname{trace}(A_kX) = b_k \quad (k = 1, \ldots, m),\; X \in \mathcal{A}_{SDP}\right\}. \qquad (9)$$

Assume now that we have an orthogonal basis $B_1, \ldots, B_d$ of $\mathcal{A}_{SDP}$. We set

$X = \sum_{i=1}^{d} x_i B_i$ to get

$$\begin{aligned}
&\min_{X \succeq 0} \{\mathrm{trace}(A_0 X) \;:\; \mathrm{trace}(A_k X) = b_k \quad (k = 1, \ldots, m), X \in \mathcal{A}_{SDP}\} \\
= &\min_{\sum_{i=1}^{d} x_i B_i \succeq 0} \left\{ \sum_{i=1}^{d} x_i \mathrm{trace}(A_0 B_i) \;:\; \sum_{i=1}^{d} x_i \mathrm{trace}(A_k B_i) = b_k, \right. \qquad (10) \\
&\left. (k = 1, \ldots, m) \right\}.
\end{aligned}$$

If $\mathcal{A}_{SDP}$ is a coherent configuration (see Definition 2), then we may assume that the B_i's are zero-one matrices that sum to the all ones matrix. In this case, adding the additional constraint $X \geq 0$ (i.e. X elementwise nonnegative) to problem (9) is equivalent to adding the additional constraint $x \geq 0$ to (10).

We may now replace the linear matrix inequality in the last SDP problem by an equivalent one,

$$\sum_{i=1}^{d} x_i B_i \succeq 0 \Longleftrightarrow \sum_{i=1}^{d} x_i Q^* B_i Q \succeq 0,$$

to get a block-diagonal structure, where Q is the unitary matrix that provides the Wedderburn decomposition of $\mathcal{A}_{SDP}$. In particular, we obtain

$$Q^* B_k Q =: \oplus_{i=1}^{t} t_i \odot B_k^{(i)} \quad (k = 1, \ldots, d)$$

for some Hermitian matrices $B_k^{(i)} \in \mathbb{C}^{n_i \times n_i}$ $(i = 1, \ldots, t)$. Subsequently, we may delete any identical copies of blocks in the block structure to obtain a final reformulation. In particular, $\sum_{i=1}^{d} x_i Q^* B_i Q \succeq 0$ becomes

$$\sum_{k=1}^{d} x_k \oplus_{i=1}^{t} B_k^{(i)} \succeq 0.$$

Thus we arrive at the final SDP reformulation:

$$\min_{x \in \mathbb{R}^d} \left\{ \sum_{i=1}^{d} x_i \mathrm{trace}(A_0 B_i) \;:\; \sum_{i=1}^{d} x_i \mathrm{trace}(A_k B_i) = b_k \; \forall k, \; \sum_{k=1}^{d} x_k \oplus_{i=1}^{t} B_k^{(i)} \succeq 0 \right\}. \tag{11}$$

Note that the numbers $\mathrm{trace}(A_k B_i)$ $(k = 0, \ldots, m,\ i = 1, \ldots, d)$ may be computed beforehand.

An alternative way to arrive at the final SDP formulation (11) is as follows. We may first replace the LMI $\sum_{i=1}^{d} x_i B_i \succeq 0$ by $\sum_{i=1}^{d} x_i L_i \succeq 0$ where the L_i's $(i = 1, \ldots, d)$ form the basis of the regular *-representation of $\mathcal{A}_{SDP}$. Now we may replace the latter LMI using the Wedderburn decomposition (block-diagonalization) of the L_i's, and delete any duplicate blocks as before. These two approaches result in the same final SDP formulation, but the latter approach offers numerical advantages and is the one used to obtain the numerical results in Section 8.

Note that, even if the data matrices A_i are real symmetric, the final block diagonal matrices in (11) may in principle be complex Hermitian matrices, since Q may be unitary (as opposed to real orthogonal). This poses no problem in theory, since interior point methods apply to SDP with Hermitian data matrices as well. If required, one may reformulate a Hermitian linear matrix inequality in terms of real matrices by applying the relation

$$A \succeq 0 \Longleftrightarrow \left[\begin{array}{cc} \mathrm{Re}(A) & \mathrm{Im}(A)^T \\ \mathrm{Im}(A) & \mathrm{Re}(A) \end{array}\right] \succeq 0 \quad (A = A^* \in \mathbb{C}^{n\times n})$$

to each block in the LMI. Note that this doubles the size of the block.

A summary of the symmetry reduction algorithm for SDP is given as Algorithm 4.

Algorithm 4 Symmetry reduction of the SDP problem (6)

```
INPUT: data for the SDP problem (6), and a real, orthonormal basis $D_1, \ldots, D_d$ of $\mathcal{A}_{SDP}$.
(i) Compute the basis $L_1, \ldots, L_d$ of $\mathcal{A}_{SDP}^{reg}$ as described in Section 5;
(ii) Obtain a generic element from center($\mathcal{A}_{SDP}^{reg}$) via Algorithm 3;
(iii) Decompose $\mathcal{A}_{SDP}^{reg}$ into simple components using Algorithm 1;
(iv) Decompose the simple components from step (iii) into basic components using Algorithm 2.
OUTPUT: The reduced SDP of the form (11).
```

We conclude this section with a few remarks on the use of Algorithm 4 in practice.

6.1 Symmetry from permutation groups

In most applications of symmetry reduction in SDP, the algebra $\mathcal{A}_{SDP}$ arises as the centralizer ring of some permutation group; see *e.g.* §4.4 in [11] or Chapter 2 in [8] for more details.

In particular, assume we have a permutation group, say $\mathcal{G}_{SDP}$, acting on $\{1, \ldots, n\}$ such that

$$(A_k)_{i,j} = (A_k)_{\sigma(i),\sigma(j)} \; \forall \sigma \in \mathcal{G}_{SDP}, \; i,j \in \{1, \ldots, n\}$$

holds for $k = 0, \ldots, m$. Let us define the permutation matrix representation of the group $\mathcal{G}_{SDP}$ as follows:

$$(P_\sigma)_{i,j} := \left\{ \begin{array}{ll} 1 & \text{if } \pi(i) = j \\ 0 & \text{else.} \end{array} \right. \qquad \sigma \in \mathcal{G}_{SDP}, \; i,j = 1, \ldots, n.$$

Now one has

$$A_k P_\sigma = P A_k \quad \forall \sigma \in \mathcal{G}_{SDP}, \; k = 0, \ldots, m,$$

i.e. the SDP data matrices commute with all the permutation matrices P_σ ($\sigma \in \mathcal{G}_{SDP}$). In other words, $A_0, \ldots, A_m$ are contained in

$$\{A \in \mathbb{C}^{n\times n} \mid AP_\sigma = P_\sigma A \quad \forall \sigma \in \mathcal{G}_{SDP}\}.$$

This set is called the commutant or *centralizer ring* of the group $\mathcal{G}_{SDP}$, or the *commuting algebra* of the group. It forms a matrix $*$-algebra over $\mathbb{C}$, as is easy to show, and we may therefore choose $\mathcal{A}_{SDP}$ to be the commutant.

In this case, $\mathcal{A}_{SDP}$ is a coherent configuration, and an orthogonal basis of $\mathcal{A}_{SDP}$ is obtained from the so-called orbitals of $\mathcal{G}_{SDP}$.

Definition 5. *The* two-orbit or orbital of an index pair (i, j) is defined as

$$\{(\sigma(i), \sigma(j)) \; : \; \sigma \in \mathcal{G}_{SDP}\}.$$

The orbitals partition $\{1, \ldots, n\} \times \{1, \ldots, n\}$ and this partition yields the $0-1$ basis matrices of the coherent configuration.

Moreover, in this case the regular $*$-representation of $\mathcal{A}_{SDP}$ may be efficiently computed; see *e.g.* [31]. In the numerical examples to be presented in Section 8, we will deal with this situation.

Finally, if the algebra $\mathcal{A}_{SDP}$ is not from a permutation group, but still contained in some (low dimensional) coherent configuration, then one may use an efficient combinatorial procedure known as *stabilization* to find this coherent configuration; see [31] for details.

7 Relation to an approach by Murota *et al.* [21]

In this section we explain the relation between our approach and the ones in Murota *et al.* [21] and Maehara and Murota [19]. In these papers the authors study matrix $*$-algebras over $\mathbb{R}$ (as opposed to $\mathbb{C}$). This is more complicated than studying matrix $\mathbb{C}*$-algebras, since there is no simple analogy of the Wedderburn decomposition theorem (Theorem 1) for matrix $*$-algebras over $\mathbb{R}$. While any simple matrix $\mathbb{C}*$-algebra is basic, there are three types of simple matrix $*$-algebras over $\mathbb{R}$; see [19] for a detailed discussion.

We therefore assume now, as in [21], that $\mathcal{A}_{SDP}$ is a matrix $*$-algebra over $\mathbb{R}$, and that

$$\mathcal{A}_{SDP} \cap \mathbb{S}^{n\times n} = \text{span}\{A_0, \ldots, A_m\},$$

and that

$$\mathcal{A}_{SDP} = \langle\{A_0, \ldots, A_m\}\rangle.$$

In words, the A_i's generate $\mathcal{A}_{SDP}$ and form a basis for the symmetric part of $\mathcal{A}_{SDP}$.

The approach of Murota *et al.* (Algorithm 4.1 in [21]) to decompose $\mathcal{A}_{SDP}$ into simple components works as follows.

Algorithm 4.1 in [21]

1. Choose a random $r = [r_0, \dots, r_m]^T \in \mathbb{R}^{m+1}$;
2. Let $A := \sum_{i=0}^m r_i A_i$;
3. Perform the spectral decomposition of A to obtain an orthogonal matrix, say Q, of eigenvectors.
4. Make a k-partition of the columns of Q that defines matrices Q_i ($i = 1, \dots, k$) so that
$$Q_i^T A_p Q_j = 0 \quad \forall\, p = 0, \dots, m, \quad i \neq j. \tag{12}$$

Similar to Algorithm 1, Algorithm 4.1 in [21] involves sampling from the center of $\mathcal{A}_{SDP}$. To prove this, we will require the following lemma.

Lemma 2. *Assume $A = A^T \in \mathcal{A}_{SDP}$ has spectral decomposition $A = \sum_i \lambda_i q_i q_i^T$. Then, for any (eigen)values $r_i \in \mathbb{R}$, the matrix*
$$\sum_i r_i q_i q_i^T$$
is also in $\mathcal{A}_{SDP}$, provided that $r_i = r_{i'}$ whenever $\lambda_i = \lambda_{i'}$.

Proof. The result of the lemma is a direct consequence of the properties of Vandermonde matrices. □

Theorem 5. *The matrices $Q_i Q_i^T$ ($i = 1, \dots, k$) are symmetric, central idempotents of $\mathcal{A}_{SDP}$.*

Proof. For each i, let $E_i := Q_i Q_i^T$, and note that $E_i^2 = E_i$, i.e. E_i is idempotent. Also note that
$$\sum_{i=1}^k E_i = QQ^T = I. \tag{13}$$
Fix $p \in \{0, \dots, m\}$. One has
$$\begin{aligned} E_i A_p E_j &= Q_i \left(Q_i^T A_p Q_j\right) Q_j^T \\ &= 0 \text{ if } i \neq j. \end{aligned}$$

By (13) one has
$$\sum_{i=1}^k E_i A_p = A_p \text{ and } \sum_{i=1}^k A_p E_i = A_p,$$
which implies $E_j A_p E_j = A_p E_j$ and $E_j A_p E_j = E_j A_p$ respectively ($j = 1, \dots, k$). Thus, $E_j A_p = A_p E_j$ for all $j = 1, \dots, k$. Since the A_i ($i = 1, \dots, m$) are generators of $\mathcal{A}_{SDP}$, this means that the E_j's ($j = 1, \dots, k$) are in the commutant of $\mathcal{A}_{SDP}$.

It remains to show that $E_j \in \mathcal{A}_{SDP}$ $(j = 1, \ldots, k)$. This follows directly from Lemma 2.

Note that E_j and A share the set Q of eigenvectors, thus $E_j \in \mathcal{A}_{SDP}$ $(j = 1, \ldots, k)$ by the lemma. Thus E_j $(j = 1, \ldots, k)$ is in the center of $\mathcal{A}_{SDP}$.

□

Note that the matrix Q is implicitly used to construct the matrices E_j;

In particular, the the k-partition of Q yields the matrices Q_j and then $E_j := Q_j Q_j^T$ are central idempotents.

8 Numerical experiments

In this section we illustrate the proposed symmetry reduction technique for SDP instances from two sources:

1. instances that give lower bounds on the crossing numbers of complete bipartite graphs.
2. The calculation of the ϑ'-number of certain graphs with large automorphism groups.

Unless otherwise indicated, all computation was done using the SDPT3 [27] solver and a Pentium IV PC with 2GB of RAM memory.

8.1 Numerical results for crossing number SDP's

Recall that the crossing number $\mathrm{cr}(G)$ of a graph G is the minimum number of intersections of edges in a drawing of G in the plane.

The crossing number of the complete bipartite graph $K_{r,s}$ is only known in a few special cases (like $\min\{r, s\} \le 6$), and it is therefore interesting to obtain lower bounds on $\mathrm{cr}(K_{r,s})$. (There is a well known upper bound on $\mathrm{cr}(K_{r,s})$ via a drawing which is conjectured to be tight.)

De Klerk *et al.* [12] showed that one may obtain a lower bound on $\mathrm{cr}(K_{r,s})$ via the optimal value of a suitable SDP problem, namely:

$$\mathrm{cr}(K_{r,s}) \ge \frac{s}{2}\left(s \min_{X \ge 0,\ X \succeq 0} \{\mathrm{trace}(QX) \mid \mathrm{trace}(JX) = 1\} - \left\lfloor \frac{r}{2} \right\rfloor \left\lfloor \frac{r-1}{2} \right\rfloor \right),$$

where Q is a certain (given) matrix of order $n = (r-1)!$, and J is the all-ones matrix of the same size. The rows and columns of Q are indexed by all the cyclic orderings of r elements. For this SDP problem the algebra $\mathcal{A}_{SDP}$ is a coherent configuration and an orthogonal basis $B_1, \ldots, B_d$ of zero-one matrices of $\mathcal{A}_{SDP}$ is available.

Some information on $\mathcal{A}_{SDP}$ is given in Table 1.

The instance corresponding to $r = 7$ was first solved in [12], by solving the partially reduced SDP problem (9), that in this case takes the form:

$$\min_{\sum_{i=1}^d x_i B_i \succeq 0, x \ge 0} \left\{ \sum_{i=1}^d x_i \mathrm{trace}(QB_i) \ : \ \sum_{i=1}^d x_i \mathrm{trace}(JB_i) = 1 \right\}.$$

r	$n=(r-1)!$	$d:=\dim(\mathcal{A}_{SDP})$	$\dim(\mathcal{A}_{SDP}\cap\mathbb{S}^{n\times n})$
7	720	78	56
8	5040	380	239
9	40320	2438	1366

Table 1: Information on $\mathcal{A}_{SDP}$ for the crossing number SDP instances

The larger instances where $r = 8, 9$ were solved in [13] by solving the equivalent, but smaller problem:

$$\min_{\sum_{i=1}^d x_i L_i \succeq 0, x\geq 0} \left\{ \sum_{i=1}^d x_i \mathrm{trace}(QB_i) \ : \ \sum_{i=1}^d x_i \mathrm{trace}(JB_i) = 1 \right\}, \tag{14}$$

where the L_i's $(i = 1, \ldots, d)$ form the basis of $\mathcal{A}_{SDP}^{reg}$.

In what follows we further reduce the latter problem by computing the Wedderburn decomposition of $\mathcal{A}_{SDP}^{reg}$ using Algorithm 4. We computed the basis of $\mathcal{A}_{SDP}^{reg}$ using a customized extension of the computational algebra package GRAPE [25], that in turn is part of the GAP routine library [6].

The Wedderburn decomposition results in block diagonalization of the L_i's $(i = 1, \ldots, d)$, and the sizes of the resulting blocks are shown in Table 2.

r	$t_i = n_i$
7	3 (6×), 2 (4×), 1 (8×)
8	7 (2×), 5 (2×), 4 (9×), 3 (7×), 2 (4×), 1 (9×),
9	12 (8×), 11 (2×), 9 (6×), 7 (3×), 6 (5×), 5 (2×), 4 (2×), 3 (16×), 1 (5×)

Table 2: The block sizes in the decomposition $Q^*\mathcal{A}_{SDP}^{reg}Q = \oplus_i t_i \odot \mathbb{C}^{n_i\times n_i}$. Since $\mathcal{A}_{SDP}^{reg}$ is the regular $*$-representation of $\mathcal{A}_{SDP}$ one has $t_i = n_i$ for all i (see Theorem 3).

The difference between the sparsity patterns of a generic matrix in $\mathcal{A}_{SDP}^{reg}$ before and after symmetry reduction is illustrated in Figure 1 when $r = 9$. In this case, $\mathcal{A}_{SDP}^{reg} \subset \mathbb{C}^{2438\times 2438}$. Before symmetry reduction, there is no discernable sparsity pattern. After $\mathcal{A}_{SDP}^{reg}$ is decomposed into simple components, a block diagonal structure is visible, with largest block size 144. After the simple components are decomposed into basic components, the largest block size is 12.

Table 3 gives the solution times for the SDP instances (14) before and after the numerical symmetry reduction (Algorithm 4).

For $r = 9$, the solution time reported in [13] was 7 days of wall clock time on a SGI Altix supercomputer. Using the numerical symmetry reduction this reduces to about 24 minutes on a Pentium IV PC, including the time for block diagonalization.

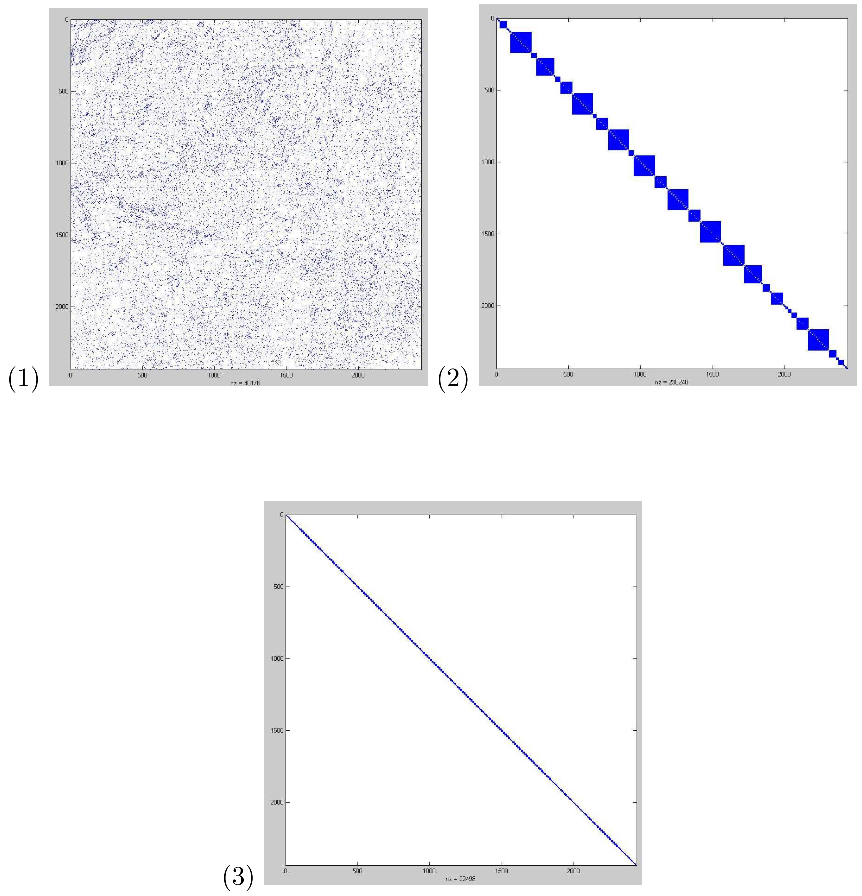

Figure 1: The sparsity pattern of $\mathcal{A}_{SDP}^{reg}$ for $r = 9$: (1) before any pre-processing, (2) after decomposition into simple components, and (3) after decomposition into basic components.

For all three instances we obtained 6 digits of accuracy in the optimal value. Moreover, the same results were obtained for different random samples in the subroutines of Algorithms 3 and 2.

The other dominant operation for the block diagonalization is executing Algorithm 3 (sampling from the center of $\mathcal{A}_{SDP}^{reg}$). The time required for this is shown in Table 3.

r	CPU time Algorithm 3	Solution time (14) after block diagonalization	Solution time (14)
9	16 min 16 sec	7 min 48 sec	> 7 days[†]
8	4.7 sec	3.2 sec	5 min 22 sec
7	0.04 sec	0.6 sec	2.7 sec

Table 3: Solution times on a Pentium IV PC for the SDP instances before and after decomposition. [†] refers to wall clock computation time on an SGI Altix supercomputer cluster.

8.2 Results for the ϑ'-number of graphs

The ϑ'-number of a graph was introduced in [20] as a strengthening of the Lovász ϑ-number [18] upper bound on the co-clique number of a graph. The ϑ'-number was also studied in detail for Hamming graphs in the seminal paper by Schrijver [23].

The ϑ'-number of a graph G with adjacency matrix A may be defined as:

$$\vartheta'(G) := \max_{X \succeq 0,\ X \geq 0} \{\mathrm{trace}(JX) \mid \mathrm{trace}((A+I)X) = 1\}. \tag{15}$$

Note that the symmetry group $\mathcal{G}_{SDP}$ of this SDP coincides with the automorphism group of the graph. Thus we may take $\mathcal{A}_{SDP}$ to be the centralizer ring of this group, as before.

In [15], the ϑ' number of the so-called Erdös-Renyi graphs was studied. These graphs, denoted by $ER(q)$. are determined by a singe parameter $q > 2$, which is prime. The number of vertices is $n = q^2 + q + 1$, but the dimension of $\mathcal{A}_{SDP}$ is only $2q + 11$.

Note that, for example, if $q = 157$, one has $n = 24807$, making it impossible to solve (15) without exploiting the symmetry.

The Wedderburn decomposition of $\mathcal{A}_{SDP}$ is not known in closed form [15], and the numerical techniques proposed in this paper may therefore be employed.

As before, we denote the zero-one basis of $\mathcal{A}_{SDP}$ by $B_1, \ldots, B_d$ and the basis of its regular *-representation $\mathcal{A}_{SDP}^{reg}$ by $L_1, \ldots, L_d$. In [15], $\vartheta'(ER(q))$ was computed, for $q \leq 31$, by solving the SDP:

$$\vartheta'(ER(q)) = \min_{\sum_{i=1}^{d} x_i L_i \succeq 0, x \geq 0} \left\{ \sum_{i=1}^{d} x_i \mathrm{trace}(JB_i) \ : \ \sum_{i=1}^{d} x_i \mathrm{trace}((A+I)B_i) = 1 \right\}, \tag{16}$$

where A denotes the adjacency matrix of the graph $ER(q)$. The times required to compute the matrices $L_1, \ldots, L_d$ are given in Table 4, and were obtained using the GRAPE [25] software, as before.

We can compute $\vartheta'(ER(q))$ for larger values of q, by obtaining the Wedderburn decomposition of $\mathcal{A}_{SDP}^{reg}$ using Algorithm 4. The resulting block sizes are also given in Table 4.

q	d	Computing $L_1, \ldots, L_d$	$t_i = n_i$
157	325	1 h 22 min	2 (79 $\times$), 3 (1 $\times$)
101	213	21 min	2 (51 $\times$), 3 (1 $\times$)
59	129	4 min	2 (30 $\times$), 3 (1 $\times$)
41	93	1 min 22 sec	2 (21 $\times$), 3 (1 $\times$)
31	73	37 sec	2 (16 $\times$), 3 (1 $\times$)

Table 4: The block sizes in the decomposition $Q^*\mathcal{A}_{SDP}^{reg}Q = \oplus_i t_i \odot \mathbb{C}^{n_i \times n_i}$ where $\mathcal{A}_{SDP}^{reg}$ is the centralizer ring of $\mathrm{Aut}(ER(q))$.

Note that the largest block size appearing in Table 4 is 3. For this reason all the ϑ' values listed in the the table were computed in a few seconds after the symmetry reduction; see Table 5.

In Table 5 the time required to perform the block diagonalization is shown in the second column (i.e. the execution time for Algorithm 3).

In the last column we also give the value of $ER(q)$, since these values have not been computed previously for $q > 31$.

q	CPU time Algorithm 3	Solution time (16) after block diagonalization	Solution time (16)	$\vartheta'(ER(q))$
157	7.47	5.5	351.8	1834.394
101	1.34	1.4	70.3	933.137
59	0.21	0.8	11.6	408.548
41	0.047	0.61	6.5	233.389
31	0.018	0.49	3.4	151.702

Table 5: Times(sec) to compute $\vartheta'(ER(q))$ with and without block diagonalization.

Note that, for $q = 157$, the block diagonalization plus the solution of the resulting SDP takes a total of $7.47 + 5.5 \approx 13$ seconds, as opposed to the 351.8 seconds required to solve (16) without block diagonalization.

8.3 Numerical comparison with approach by Murota *et al.* [21]

Finally, we evaluated the algorithm by Murota *et al.* (Algorithm 4.1 in [21], see Section 7) for the SDP instances studied here.

Thus Algorithm 4.1 in [21] (see Section 7) is used to block diagonalize $\mathcal{A}_{SDP}^{reg}$ into simple components.

Algorithm 4.1 in [21] produced the same block diagonal structure as shown in Tables 2 and 4 when combined with Algorithm 2.

Note however, that this identical behavior is not guaranteed in general, since Algorithm 4.1 in [21] applies to matrix algebras over the reals. For the examples studied here though, one could replace Algorithm 3 by Algorithm 4.1 in [21] in our Algorithm 4.

Acknowledgements

The authors are deeply indebted to Professor Masakazu Kojima for suggesting the outline of the proof of Theorem 4, and for making his Matlab implementation of Algorithm 4.1 in [21] available to them.

The authors also thank an anonymous referee for pointing out erroneous numerical results in an earlier version of this paper.

References

[1] F. Alizadeh. *Combinatorial optimization with interior point methods and semi–definite matrices.* PhD thesis, University of Minnesota, Minneapolis, USA, 1991.

[2] C. Bachoc and F. Vallentin. New upper bounds for kissing numbers from semidefinite programming. *Journal of the AMS*, **21**:909-924, 2008.

[3] Y.-Q. Bai, E. de Klerk, D.V. Pasechnik, and R. Sotirov. Exploiting group symmetry in truss topology optimization. *Optimization and Engineering* **10**(3):331–349, 2009.

[4] P.J. Cameron. Coherent configurations, association schemes, and permutation groups, pp. 55-71 in Groups, Combinatorics and Geometry (A. A. Ivanov, M. W. Liebeck and J. Saxl eds.), World Scientific, Singapore, 2003.

[5] W. Eberly and M. Giesbrecht. Efficient decomposition of separable algebras. *Journal of Symbolic Computation*, 37(1):35–81, 2004.

[6] GAP – Groups, Algorithms, and Programming, Version 4.4.12, The GAP Group, 2008, `http://www.gap-system.org`

[7] K. Gatermann and P.A. Parrilo, Symmetry groups, semidefinite programs, and sums of squares. *J. Pure and Applied Algebra*, **192**:95–128, 2004.

[8] D. Gijswijt. Matrix Algebras and Semidefinite Programming Techniques for Codes. PhD thesis, University of Amsterdam, The Netherlands, 2005. `http://staff.science.uva.nl/~gijswijt/promotie/thesis.pdf`

[9] D. Gijswijt, A. Schrijver, H. Tanaka, New upper bounds for nonbinary codes based on the Terwilliger algebra and semidefinite programming, *Journal of Combinatorial Theory, Series A*, **113**:1719–1731, 2006.

[10] Y. Kanno, M. Ohsaki, K. Murota and N. Katoh. Group symmetry in interior-point methods for semidefinite program, *Optimization and Engineering*, **2**(3): 293–320, 2001.

[11] E. de Klerk. Exploiting special structure in semidefinite programming: a survey of theory and applications. *European Journal of Operational Research*, **201**(1): 1–10, 2010.

[12] E. de Klerk, J. Maharry, D.V. Pasechnik, B. Richter and G. Salazar. Improved bounds for the crossing numbers of $K_{m,n}$ and K_n. *SIAM Journal on Discrete Mathematics* **20**:189–202, 2006.

[13] E. de Klerk, D.V. Pasechnik and A. Schrijver. Reduction of symmetric semidefinite programs using the regular *-representation. *Mathematical Programming B*, **109**(2-3):613-624, 2007.

[14] E. de Klerk and R. Sotirov. Exploiting Group Symmetry in Semidefinite Programming Relaxations of the Quadratic Assignment Problem. *Mathematical Programming, Series A*, **122**(2):225–246, 2010.

[15] E. de Klerk, M.W. Newman, D.V. Pasechnik, and R. Sotirov. On the Lovász ϑ-number of almost regular graphs with application to Erdös-Rényi graphs. *European Journal of Combinatorics*, **31**:879–888, 2009.

[16] M. Kojima, S. Kojima and S. Hara. Linear algebra for semidefinite programming, Research Report B-290, Tokyo Institute of Technology, October 1994; also in RIMS Kokyuroku 1004, Kyoto University, pp. 1-23, 1997.

[17] M. Laurent. Strengthened semidefinite bounds for codes. *Mathematical Programming*, **109**(2-3):239–261, 2007.

[18] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information theory*, **25**:1–7, 1979.

[19] T. Maehara and K. Murota. A numerical algorithm for block-diagonal decomposition of matrix *-algebras, Part II: general algorithm, May 2009, submitted for publication. (Original version: A numerical algorithm for block-diagonal decomposition of matrix *-algebras with general irreducible components, METR 2008-26, University of Tokyo, May 2008.)

[20] R.J. McEliece, E.R. Rodemich, and H.C. Rumsey (1978), The Lovász bound and some generalizations. *Journal of Combinatorics, Information & System Sciences* **3**:134152, 1978.

[21] K. Murota, Y. Kanno, M. Kojima and S. Kojima, A Numerical Algorithm for Block-Diagonal Decomposition of Matrix *-Algebras, Technical report B-445, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, September 2007 (Revised June 2008).

[22] Yu.E. Nesterov and A.S. Nemirovski. *Interior point polynomial algorithms in convex programming.* SIAM Studies in Applied Mathematics, Vol. 13. SIAM, Philadelphia, USA, 1994.

[23] A. Schrijver. A comparison of the Delsarte and Lovász bounds. *IEEE Transactions on Information Theory*, **25**:425–429, 1979.

[24] A. Schrijver. New code upper bounds from the Terwilliger algebra. *IEEE Transactions on Information Theory*, **51**:2859–2866, 2005.

[25] L.H. Soicher. GRAPE: GRaph Algorithms using PErmutation groups—A GAP package, Version number 4.3, 2006, `http://www.maths.qmul.ac.uk/~leonard/grape/`

[26] M.J. Todd. Semidefinite optimization. Acta Numerica, **10**:515–560, 2001.

[27] K. C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3—a MATLAB software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, **11/12**(1-4):545–581, 1999.

[28] F. Vallentin. Symmetry in semidefinite programs. *Linear Algebra and its Applications*, to appear. Preprint version: arXiv:0706.4233

[29] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, **38**:49–95, 1996.

[30] J.H.M. Wedderburn. On hypercomplex numbers. *Proc. London Math. Soc.* **6**(2):77–118, 1907.

[31] B. Weisfeiler, editor. *On Construction and Identification of Graphs*, Springer, Lecture Notes in Mathematics 558, 1976.