

# SEMIDEFINITE PROGRAMMING AND SUMS OF HERMITIAN SQUARES OF NONCOMMUTATIVE POLYNOMIALS

IGOR KLEP<sup>1</sup> AND JANEZ POVH<sup>2</sup>

ABSTRACT. An algorithm for finding sums of hermitian squares decompositions for polynomials in noncommuting variables is presented. The algorithm is based on the “Newton chip method”, a noncommutative analog of the classical Newton polytope method, and semidefinite programming.

## 1. INTRODUCTION

**1.1. Notation.** We write  $\mathbb{N} := \{1, 2, \dots\}$ ,  $\mathbb{R}$  for the sets of natural and real numbers. Let  $\langle \bar{X} \rangle$  be the monoid freely generated by  $\bar{X} := (X_1, \dots, X_n)$ , i.e.,  $\langle \bar{X} \rangle$  consists of *words* in the  $n$  noncommuting letters  $X_1, \dots, X_n$  (including the empty word denoted by 1).

We consider the algebra  $\mathbb{R}\langle \bar{X} \rangle$  of polynomials in  $n$  noncommuting variables  $\bar{X} = (X_1, \dots, X_n)$  with coefficients from  $\mathbb{R}$ . The elements of  $\mathbb{R}\langle \bar{X} \rangle$  are linear combinations of words in the  $n$  letters  $\bar{X}$  and are called *NC polynomials*. The length of the longest word in an NC polynomial  $f \in \mathbb{R}\langle \bar{X} \rangle$  is the *degree* of  $f$  and is denoted by  $\deg f$ . We shall also consider the degree of  $f$  in  $X_i$ ,  $\deg_i f$ . Similarly, the length of the shortest word appearing in  $f \in \mathbb{R}\langle \bar{X} \rangle$  is called the *min-degree* of  $f$  and denoted by  $\text{mindeg } f$ . Likewise,  $\text{mindeg}_i f$  is introduced. If the variable  $X_i$  does not occur in some monomial in  $f$ , then  $\text{mindeg}_i f = 0$ . For instance, if  $f = X_1^3 + 2X_1X_2X_3 - X_1^2X_4^2$ , then

$$\deg f = 4, \quad \deg_1 f = 3, \quad \deg_2 f = \deg_3 f = 1, \quad \deg_4 f = 2,$$

$$\text{mindeg } f = 3, \quad \text{mindeg}_1 f = 1, \quad \text{mindeg}_2 f = \text{mindeg}_3 f = \text{mindeg}_4 f = 0.$$

An element of the form  $aw$  where  $0 \neq a \in \mathbb{R}$  and  $w \in \langle \bar{X} \rangle$  is called a *monomial* and  $a$  its *coefficient*. Hence words are monomials whose coefficient is 1.

---

*Date:* July 18, 2009.

*2000 Mathematics Subject Classification.* Primary 11E25, 90C22; Secondary 08B20, 13J30.

*Key words and phrases.* noncommutative polynomial, sum of squares, semidefinite programming, Newton polytope.

<sup>1</sup>Partially supported by the Slovenian Research Agency (project no. Z1-9570-0101-06).

<sup>2</sup>Supported by the Slovenian Research Agency (project no. 1000-08-210518).

We equip  $\mathbb{R}\langle\bar{X}\rangle$  with the *involution*  $*$  that fixes  $\mathbb{R} \cup \{\bar{X}\}$  pointwise and thus reverses words, e.g.

$$(X_1X_2 - X_1^2X_3)^* = X_2X_1 - X_3X_1^2.$$

Hence  $\mathbb{R}\langle\bar{X}\rangle$  is the  $*$ -algebra freely generated by  $n$  symmetric letters. Let  $\text{Sym } \mathbb{R}\langle\bar{X}\rangle$  denote the set of all *symmetric elements*, that is,

$$\text{Sym } \mathbb{R}\langle\bar{X}\rangle = \{f \in \mathbb{R}\langle\bar{X}\rangle \mid f = f^*\}.$$

An NC polynomial of the form  $g^*g$  is called a *hermitian square* and the set of all sums of hermitian squares will be denoted by  $\Sigma^2$ . Clearly,  $\Sigma^2 \subsetneq \text{Sym } \mathbb{R}\langle\bar{X}\rangle$ . For example,

$$\begin{aligned} X_1X_2 - X_2X_1 &\notin \text{Sym } \mathbb{R}\langle\bar{X}\rangle, & X_1X_2X_1 &\in \text{Sym } \mathbb{R}\langle\bar{X}\rangle \setminus \Sigma^2, \\ 1 - 2X_1 + 2X_1^2 + X_1X_2 + X_2X_1 - X_1^2X_2 - X_2X_1^2 + X_2X_1^2X_2 &= \\ &= (1 - X_1 + X_1X_2)^*(1 - X_1 + X_1X_2) + X_1^2 \in \Sigma^2. \end{aligned}$$

The involution  $*$  extends naturally to matrices (in particular, to vectors) over  $\mathbb{R}\langle\bar{X}\rangle$ . For instance, if  $V = (v_i)$  is a (column) vector of NC polynomials  $v_i \in \mathbb{R}\langle\bar{X}\rangle$ , then  $V^*$  is the row vector with components  $v_i^*$ . We shall also use  $V^t$  to denote the row vector with components  $v_i$ .

Occasionally one needs to work with the free  $*$ -algebra  $\mathbb{R}\langle\bar{X}, \bar{X}^*\rangle$ , i.e., the free  $*$ -algebra freely generated by  $n$  (nonsymmetric) NC variables  $\bar{X}$ , or with the mixed case where some of the variables are symmetric and some are not. All of the notions introduced above in the case of symmetric variables have natural counterparts in  $\mathbb{R}\langle\bar{X}, \bar{X}^*\rangle$ . For the sake of exposition, we have restricted ourselves to  $\mathbb{R}\langle\bar{X}\rangle$  but the interested reader will have no problems adapting the results to  $\mathbb{R}\langle\bar{X}, \bar{X}^*\rangle$ .

**1.2. Motivation and Contribution.** If  $f \in \mathbb{R}\langle\bar{X}\rangle$  is a sum of hermitian squares and we substitute self-adjoint matrices  $A_1, \dots, A_n$  of the same size for the variables  $\bar{X}$ , then the resulting matrix  $f(A_1, \dots, A_n)$  is positive semidefinite. Recall that a matrix  $A$  is called *positive semidefinite* if it is self-adjoint and  $\langle Av, v \rangle \geq 0$  for all vectors  $v$ . Equivalently:  $A$  is self-adjoint and all of its eigenvalues are nonnegative. For self-adjoint matrices  $A$  and  $B$  of the same size, we write  $A \succeq B$  to express that  $A - B$  is positive semidefinite, i.e.,

$$A \succeq B \Leftrightarrow \langle Av, v \rangle \geq \langle Bv, v \rangle \text{ for all vectors } v.$$

Helton [Hel02] proved (a slight variant of) the converse of the above observation: If  $f \in \mathbb{R}\langle\bar{X}\rangle$  and  $f(A_1, \dots, A_n) \succeq 0$  for *all* self-adjoint matrices  $A_i$  of the same size, then  $f$  is a sum of hermitian squares. For a beautiful exposition, we refer the reader to [MP05].

Together with coworkers Helton pursued this line of research further, studied positivity and convexity of NC polynomials and gave applications to control theory, optimization, systems engineering, etc.; see [dOHMP08] for a nice survey of these beginnings of *free semialgebraic geometry*. The first author in [KS08a] connected sums of hermitian squares of NC polynomials to an old open problem of Connes on von Neumann algebras, and, somewhat related, found applications to mathematical physics [KS08b]. Many of these results were obtained with the aid of computer programs written in an ad-hoc manner.

Despite the fast rise of free semialgebraic geometry, there seems to be no published account or software implementation of an “optimal” algorithm for computing (or determining existence) of sums of hermitian squares (SOHS) decompositions of NC polynomials. The main contribution of this article is to present such an algorithm. We review the Gram matrix method to determine whether a given NC polynomial has a SOHS decomposition and how it naturally relates to semidefinite programming (SDP), see §2. In the Gram matrix method the size of the underlying semidefinite program grows very fast (exponentially with the degree of the NC polynomial) although the number of monomials one actually needs in the SOHS decomposition is always *polynomial* in the degree of the NC polynomial and the number of monomials in the NC polynomial. More precisely, for a given  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$ , at most  $\frac{k \deg f}{2}$  monomials are needed, where  $k$  is the number of symmetric monomials in  $f$ . This reduction is presented in §3 and called the *Newton chip method*. We shall also demonstrate that our method is tight, i.e., gives the exact number of monomials for some NC polynomials. Finally, in §4, §5 we discuss the duality properties of the constructed semidefinite programs and give an application of SOHS decompositions.

## 2. SUMS OF HERMITIAN SQUARES

**2.1. Sums of hermitian squares and Gram matrices.** The core of the Gram matrix method is given by the following proposition (cf. [Hel02, §2.2] or [MP05, Theorem 2.1]), the noncommutative version of the classical result due to Choi, Lam and Reznick ([CLR95, §2]; see also [Par03, PW98]). The easy proof is included for the sake of completeness.

**Proposition 2.1.** *Suppose  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  is of degree  $\leq 2d$ . Then  $f \in \Sigma^2$  if and only if there exists a positive semidefinite matrix  $G$  satisfying*

$$f = W_d^* G W_d, \tag{1}$$

where  $W_d$  is a vector consisting of all words in  $\langle \bar{X} \rangle$  of degree  $\leq d$ .

Conversely, given such a positive semidefinite matrix  $G$  with rank  $r$ , one can construct NC polynomials  $g_1, \dots, g_r \in \mathbb{R}\langle X \rangle$  of degree  $\leq d$  such that

$$f = \sum_{i=1}^r g_i^* g_i. \quad (2)$$

The matrix  $G$  is called a *Gram matrix* for  $f$ .

*Proof.* If  $f = \sum_i g_i^* g_i \in \Sigma^2$ , then  $\deg g_i \leq d$  for all  $i$  as the highest degree terms cannot cancel. Indeed, otherwise by extracting all the appropriate highest degree terms  $h_i$  with degree  $> d$  from the  $g_i$  we would obtain  $h_i \in \mathbb{R}\langle X \rangle \setminus \{0\}$  satisfying

$$\sum_i h_i^* h_i = 0. \quad (3)$$

By substituting self-adjoint matrices for variables in (3), we see that each  $h_i$  vanishes for all these substitutions. But then the nonexistence of (dimension-free) polynomial identities for tuples of self-adjoint matrices (cf. [Row80, §2.5, §1.4]) implies  $h_j = 0$  for all  $j$ . Contradiction.

Hence we can write  $g_i = G_i^t W_d$ , where  $G_i^t$  is the (row) vector consisting of the coefficients of  $g_i$ . Then  $g_i^* g_i = W_d^* G_i G_i^t W_d$  and setting  $G := \sum_i G_i G_i^t$ , (1) clearly holds.

Conversely, given a positive semidefinite  $G \in \mathbb{R}^{N \times N}$  of rank  $r$  satisfying (1), write  $G = \sum_{i=1}^r G_i G_i^t$  for  $G_i \in \mathbb{R}^{N \times 1}$ . Defining  $g_i := G_i^t W_d$  yields (2).  $\blacksquare$

**Example 2.2.** In this example we consider NC polynomials in 2 variables which we denote by  $X, Y$ . Let

$$f = 1 - 2X + X^2 + X^4 + Y^2 + Y^4 - XY^3 + X^3Y + YX^3 - Y^3X + XY^2X + YX^2Y.$$

A Gram matrix for  $f$  is given by

$$G = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix},$$

if the word vector is

$$W_2 = [1 \quad X \quad Y \quad X^2 \quad XY \quad YX \quad Y^2]^t.$$

$G$  is positive semidefinite as is easily seen from its characteristic polynomial or by observing that  $G = C^t C$  for

$$C = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}.$$

From

$$CW_2 = [1 - X \quad Y \quad X^2 + XY \quad YX - Y^2]^t$$

it follows that

$$f = (1 - X)^2 + Y^2 + (X^2 + XY)^*(X^2 + XY) + (YX - Y^2)^*(YX - Y^2) \in \Sigma^2.$$

Note that in this example all monomials from  $W_2$  appear in the SOHS decomposition of  $f$ .

Another Gram matrix for  $f$  is given by

$$G = \begin{bmatrix} 1 & -1 & 0 & \frac{1}{2} & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 1 \end{bmatrix},$$

is obviously *not* positive semidefinite and hence does not give rise to a SOHS decomposition.

**Proposition 2.3.** *Suppose  $h \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  is homogeneous of degree  $2d$  and let  $V_d$  be a vector consisting of all words in  $\langle \bar{X} \rangle$  of degree  $d$ .*

(a)  *$h$  has essentially a unique Gram matrix, i.e., there is a unique self-adjoint matrix  $G$  satisfying*

$$h = V_d^* G V_d. \quad (4)$$

(b)  *$h \in \Sigma^2$  if and only if  $G$  in (4) is positive semidefinite.*

*Proof.* (a) follows from the fact that every word of degree  $2d$  can be written *uniquely* as a product of two words of degree  $d$ .

For (b) suppose  $h \in \Sigma^2$ . In a sum of hermitian squares decomposition of  $h$  we may leave out all monomials of degree  $\neq d$  (the lowest, resp. highest degree terms cannot cancel), hence a desired positive semidefinite  $G$  exists (cf. proof of Proposition 2.1). The converse is obvious.  $\blacksquare$

For an arbitrary  $f \in \mathbb{R}\langle \bar{X} \rangle$  the Gram matrix is *not* unique, hence determining whether  $f \in \Sigma^2$  amounts to finding *a* positive semidefinite Gram matrix from the affine set of all self-adjoint Gram matrices for  $f$ . Problems like this can be (in theory) solved *exactly* using quantifier elimination [BPR06] as has been suggested in the commutative case by Powers and Wörmann [PW98]. However, this only works for problems of small size, so a *numerical* approach is needed in practice. Thus we turn to semidefinite programming.

**2.2. Semidefinite programming.** Semidefinite programming (SDP) is a subfield of convex optimization concerned with the optimization of a linear objective function over the intersection of the cone of positive semidefinite matrices with an affine space. More precisely, given self-adjoint matrices  $C, A_1, \dots, A_m$  of the same size over  $\mathbb{R}$  and a vector  $b \in \mathbb{R}^m$ , we formulate a *semidefinite program in*

*standard primal form* (in the sequel we refer to problems of this type by PSDP) as follows:

$$\begin{aligned} \inf \quad & \langle C, G \rangle \\ \text{s. t.} \quad & \langle A_i, G \rangle = b_i, \quad i = 1, \dots, m \\ & G \succeq 0. \end{aligned} \tag{PSDP}$$

Here  $\langle \cdot, \cdot \rangle$  stands for the standard scalar product of matrices:  $\langle A, B \rangle = \text{tr}(B^* A)$ . The dual problem to PSDP is the *semidefinite program in the standard dual form*

$$\begin{aligned} \sup \quad & \langle b, y \rangle \\ \text{s. t.} \quad & \sum_i y_i A_i \preceq C. \end{aligned} \tag{DSDP}$$

Here  $y \in \mathbb{R}^m$  and the difference  $C - \sum_i y_i A_i$  is usually denoted by  $Z$ .

The importance of semidefinite programming was spurred by the development of practically efficient methods to obtain (weakly) optimal solutions. More precisely, given an  $\varepsilon > 0$  we can obtain by interior point methods an  $\varepsilon$ -optimal solution with polynomially many iterations, where each iteration takes polynomially many real number operations, provided that both PSDP and DSDP have non-empty interiors of feasible sets and we have good initial points. The variables appearing in these polynomial bounds are the size  $s$  of the matrix variable, the number  $m$  of linear constraints in PSDP and  $\log \varepsilon$  (cf. [WSV00, Ch. 10.4.4]).

Note, however, that the complexity to obtain (strong) solutions of PSDP or DSDP is still a fundamental open question in semidefinite optimization [PK97]. The difficulties arise from the fact that semidefinite programs with rational input data may have irrational optimal value or/and optimal solution which are doubly exponential, hence have exponential length in any numerical system coding. Ramana [Ram97] proved that the decision problem whether there exists a feasible solution of PSDP or DSDP - the so-called SDP feasibility problem FSDP - is neither in NP nor in co-NP unless  $\text{NP} = \text{co-NP}$ , if we consider the Turing machine complexity models, and FSDP is in  $\text{NP} \cap \text{co-NP}$ , if we consider the real number model. For more details about the complexity bounds for linear, semidefinite programming and other convex quadratic programming problems we refer the reader to [BTN01].

There exist several open source packages which can efficiently find  $\varepsilon$ -optimal solutions in practice for most of the problems. If the problem is of medium size (i.e.,  $s \leq 1000$  and  $m \leq 10.000$ ), these packages are based on interior point methods, while packages for larger semidefinite programs use some variant of the first order methods (see [Mit09] for a comprehensive list of state of the art SDP solvers and also [PRW06, MPRW09]). Nevertheless, once  $s \geq 3000$  or  $m \geq 250000$ , the problem must share some special property otherwise state-of-the art solvers will fail to solve it for complexity reasons.

**2.3. Sums of hermitian squares and SDP.** In this subsection we present a *conceptual algorithm* based on SDP for checking whether  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  is a sum of hermitian squares. Following Proposition 2.1 we must determine whether there exists a positive semidefinite matrix  $G$  such that  $f = W_d^* G W_d$ , where  $W_d$  is the vector of all words of degree  $\leq d$ . This is a semidefinite feasibility problem in the matrix variable  $G$ , where the constraints  $\langle A_i, G \rangle = b_i$  are implied by the fact that for each product of monomials  $w \in W_{2d} = \{p^*q \mid p, q \in W_d\}$ ,

$$\sum_{\substack{p, q \in W_d \\ p^*q = w}} G_{p, q} = a_w, \quad (5)$$

where  $a_w$  is the coefficient of  $w$  in  $f$  ( $a_w = 0$  if the monomial  $w$  does not appear in  $f$ ).

INPUT:  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  with  $\deg f \leq 2d$ ,  $f = \sum_{w \in \langle \bar{X} \rangle} a_w w$ , where  $a_w \in \mathbb{R}$ .  
 STEP 1: Construct  $W_d$ .  
 STEP 2: Construct data  $A_i, b, C$  corresponding to the SDP.  
 STEP 3: Solve the SDP to obtain  $G$ . If the SDP is not feasible, then  $f \notin \Sigma^2$ ; stop.  
 STEP 4: Compute the Cholesky decomposition  $G = R^*R$ .  
 OUTPUT: Sum of hermitian squares decomposition of  $f$ :  $f = \sum_i g_i^* g_i$ , where  $g_i$  denotes the  $i$ -th component of  $RW_d$ .

Algorithm 1: The Gram matrix method for finding SOHS decompositions

Sums of hermitian squares and  $f$  are symmetric, and two symmetric polynomials are equal if and only if all of their ‘‘symmetrized coefficients’’ (i.e.,  $a_w + a_{w^*}$ ) coincide, hence equations (5) can be rewritten as

$$\sum_{\substack{u, v \in W_d \\ u^*v = w}} G_{u, v} + \sum_{\substack{u, v \in W_d \\ v^*u = w^*}} G_{v, u} = a_w + a_{w^*} \quad \forall w \in W_{2d}, \quad (6)$$

or equivalently,

$$\langle A_w, G \rangle = a_w + a_{w^*} \quad \forall w \in W_{2d}, \quad (7)$$

where  $A_w$  is the self-adjoint matrix defined by

$$(A_w)_{u, v} = \begin{cases} 2; & \text{if } u^*v \in \{w, w^*\}, w^* = w, \\ 1; & \text{if } u^*v \in \{w, w^*\}, w^* \neq w, \\ 0; & \text{otherwise.} \end{cases}$$

As we are interested in an arbitrary positive semidefinite  $G = [G_{u, v}]_{u, v \in W}$  satisfying the constraints (7), we can choose the objective function freely. However,

in practice one prefers solutions of small rank leading to shorter SOHS decompositions. Hence we minimize the trace, a commonly used heuristic for matrix rank minimization (cf. [RFP07]). Therefore our SDP in the primal form is as follows:

$$\begin{aligned} \inf \quad & \langle I, G \rangle \\ \text{s. t.} \quad & \langle A_w, G \rangle = a_w + a_{w^*} \quad \forall w \in W_{2d} \\ & G \succeq 0. \end{aligned} \quad (\text{SOHS}_{\text{SDP}})$$

**Remark 2.4.** The size of  $G$  in  $(\text{SOHS}_{\text{SDP}})$  is

$$N(n, d) := \sum_{k=0}^d n^k = \frac{n^{d+1} - 1}{n - 1}.$$

Thus  $N(n, d)$  grows exponentially with the polynomial degree  $d$  and easily exceeds the size manageable by the state of the art SDP solvers, which is widely accepted to be of order 1000. This implies, for example, that the above algorithm can only handle NC polynomials in two variables if they are of degree  $< 10$ . As we point out later, our method is able to work with much larger NC polynomials.

**Example 2.5.** Let

$$f = X^2 - X^{10}Y^{20}X^{11} - X^{11}Y^{20}X^{10} + X^{10}Y^{20}X^{20}Y^{20}X^{10}. \quad (8)$$

The size of a Gram matrix  $G$  for  $f$  from Proposition 2.1 is  $2^{41} - 1$  and is too big for today's SDP solvers. On the other hand, it is easy to see that

$$f = (X - X^{10}Y^{20}X^{10})^*(X - X^{10}Y^{20}X^{10}) \in \Sigma^2.$$

The polynomial  $f$  is sparse and an improved SDP for testing whether (sparse) polynomials are sums of hermitian squares will be given below.

The complexity of solving an SDP is also determined by the number of equations (7), which we denote by  $m$ . There are exactly

$$m = \text{card}\{w \in W_{2d} \mid w^* = w\} + \frac{1}{2} \text{card}\{w \in W_{2d} \mid w^* \neq w\}$$

such equations. Since  $W_d$  contains all words in  $\langle \bar{X} \rangle$  of degree  $\leq d$ , we have  $m > \frac{1}{2}N(n, 2d) = \frac{n^{2d+1}-1}{2(n-1)}$ .

For each product  $p^*q \in W_{2d}$  there are  $t$  different pairs  $(p_i, q_i)$  such that  $p_i^*q_i = p^*q$ , where  $t = \deg(p^*q) + 1$  if  $\deg(p^*q) \leq d$ , and  $t = 2d + 1 - \deg(p^*q)$  if  $\deg(p^*q) \geq d + 1$ . Note that  $t \leq d + 1$ . Therefore the matrices  $A_i$  defining the constraints (7) have order  $N(n, d)$  and every matrix  $A_i$  has at most  $d + 1$  nonzero entries, if it corresponds to a symmetric monomial of  $f$  and has at most  $2(d + 1)$  nonzero entries otherwise. Hence the matrices  $A_i$  are sparse. They are also pairwise orthogonal with respect to the scalar product  $\langle \cdot, \cdot \rangle$ , and even have disjoint supports, as we now proceed to show:

**Theorem 2.6.** *Let  $\{A_i \mid i = 1, \dots, m\}$  be the matrices constructed in Step 2 of Algorithm 1. If  $(A_i)_{r,s} \neq 0$ , then  $(A_j)_{r,s} = 0$  for all  $i \neq j$ . In particular,  $\langle A_i, A_j \rangle = 0$  for  $i \neq j$ .*

*Proof.* The equations in the SDP underlying the SOHS decomposition represent the constraints that the monomials in  $W_{2d}$  must have coefficients prescribed by the polynomial  $f$ . Let us fix  $i \neq j$ . The matrices  $A_i$  and  $A_j$  correspond to some monomials  $p_1^*q_1$  and  $p_2^*q_2$  ( $p_i, q_i \in W_d$ ), respectively, and  $p_1^*q_1 \neq p_2^*q_2$ . If  $A_i$  and  $A_j$  both have a nonzero entry at position  $(r, s)$ , then  $p_1^*q_1 = r^*s = p_2^*q_2$ , clearly a contradiction.  $\blacksquare$

**Remark 2.7.** Sparsity and orthogonality of the constraints imply that the state of the art SDP solvers can handle about 100 000 such constraints (see e.g. [MPRW09]), if the size of the matrix variable is about 1000. The boundary point method introduced in [PRW06] and analyzed in [MPRW09] has turned out to perform best for semidefinite programs of this type. It is able to use the *orthogonality* of the matrices  $A_i$  (though not the *disjointness* of their supports). In the computationally most expensive steps - solving a linear system - the system matrix becomes diagonal, so solving the system amounts to dividing by the corresponding diagonal entries.

Since  $W_d$  contains all words in  $\langle \bar{X} \rangle$  of degree  $\leq d$ , we have e.g. for  $n = 2$ ,  $d = 10$  that  $m = N(2, 20) = 2097151$  and this is clearly out of reach for *all* current SDP solvers. Nevertheless, we show in the sequel that one can replace the vector  $W_d$  in Step 1 of Algorithm 1 by a vector  $W$ , which is usually much smaller and has at most  $\frac{kd}{2}$  words, where  $k$  is the number of symmetric monomials in  $f$  and  $d = \deg f$ . Hence the size of the matrix variable  $G$  and the number of linear constraints  $m$  end up being much smaller in general.

### 3. NEWTON CHIP METHOD

We present a modification of (Step 1 of) the Gram matrix method (Algorithm 1) by implementing the appropriate noncommutative analogue of the classical Newton polytope method [Rez78], which we call the *Newton chip method*.

Define the *right chip function*  $\text{rc} : \langle \bar{X} \rangle \times \mathbb{N}_0 \rightarrow \langle \bar{X} \rangle$  by

$$\text{rc}(w_1 \cdots w_n, i) := w_{n-i+1}w_{n-i+2} \cdots w_n$$

if  $i \leq n$  and  $\text{rc}(w, i) = w$  otherwise. (In case  $i = 0$ , the empty product is defined to be the empty word 1.) As an example,  $\text{rc}(X_1X_2X_1X_2^2X_1, 4) = X_1X_2^2X_1$ .

Algorithm 2 below (the Newton chip method) reduces the word vector needed in the Gram matrix test for a sum of hermitian squares decomposition of a symmetric NC polynomial  $f$ .

INPUT:  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  with  $\deg f \leq 2d$ ,  $f = \sum_{w \in \langle \bar{X} \rangle} a_w w$ , where  $a_w \in \mathbb{R}$ .

STEP 1: Define the support of  $f$  as  $\mathcal{W}_f := \{w \in \langle \bar{X} \rangle \mid a_w \neq 0\}$ .

STEP 2:  $W := \emptyset$ .

STEP 3: Let  $m_i := \frac{\min \deg_i f}{2}$ ,  $M_i := \frac{\deg_i f}{2}$ ,  $m := \frac{\min \deg f}{2}$ ,  $M := \frac{\deg f}{2}$ . The set of admissible words is defined as

$$\mathcal{D} := \{w \in \langle \bar{X} \rangle \mid m_i \leq \deg_i w \leq M_i \text{ for all } i, m \leq \deg w \leq M\}.$$

STEP 4: For every  $w^*w \in \mathcal{W}_f$ :

Substep 4.1 For  $0 \leq i \leq \deg w$ : if  $\text{rc}(w, i) \in \mathcal{D}$ , then  $W := W \cup \{\text{rc}(w, i)\}$ .

OUTPUT:  $W$ .

Algorithm 2: The Newton chip method

**Theorem 3.1.** *Suppose  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$ . Then  $f \in \Sigma^2$  if and only if there exists a positive semidefinite  $G$  satisfying*

$$f = W^*GW, \quad (9)$$

where  $W$  is the output in vector form given by the Newton chip method.

*Proof.* Suppose  $f \in \Sigma^2$ . In every sum of hermitian squares decomposition

$$f = \sum_i g_i^* g_i, \quad (10)$$

only words from  $\mathcal{D}$  are used, i.e.,  $g_i \in \text{span } \mathcal{D}$  for every  $i$ . This follows from the fact that the lowest and highest degree terms cannot cancel (cf. proof of Proposition 2.1). Let  $\mathcal{W} := \bigcup_i \mathcal{W}_{g_i}$  be the union of the supports of the  $g_i$ . We shall prove that  $\mathcal{W} \subseteq W$ .

Let us introduce a partial ordering on  $\langle \bar{X} \rangle$ :

$$w_1 \preceq w_2 \Leftrightarrow \exists i \in \mathbb{N}_0 : \text{rc}(w_2, i) = w_1.$$

Note:  $w_1 \preceq w_2$  if and only if there is a  $v \in \langle \bar{X} \rangle$  with  $w_2 = vw_1$ .

CLAIM: For every  $w \in \mathcal{W}$  there exists  $u \in \langle \bar{X} \rangle$ :  $w \preceq u \preceq u^*u \in \mathcal{W}_f$ .

*Proof:* Clearly,  $w^*w$  is a word that appears in the representation of  $g_i^*g_i$  one naturally gets by multiplying out without simplifying, for some  $i$ . If  $w^*w \notin \mathcal{W}_f$ , then there are  $w_1, w_2 \in \mathcal{W} \setminus \{w\}$  with  $w_1^*w_2 = w^*w$  (appearing with a negative coefficient so as to cancel the  $w^*w$  term). Then  $w \preceq w_1$  or  $w \preceq w_2$ , without loss of generality,  $w \preceq w_1$ . Continuing the same line of reasoning, but starting with  $w_1^*w_1$ , we eventually arrive at  $w_\ell \in \mathcal{W}$  with  $w_\ell^*w_\ell \in \mathcal{W}_f$  and  $w \preceq w_1 \preceq \dots \preceq w_\ell$ . Thus  $w \preceq w_\ell \preceq w_\ell^*w_\ell \in \mathcal{W}_f$ , concluding the proof of the claim.

The theorem follows easily now. Since  $w_\ell^*w_\ell \in \mathcal{W}_f$  and  $w$  is a right chip of  $w_\ell$ ,  $w \in W$ . ■

**Example 3.2** (Example 2.5 continued). Applying the Newton chip method to  $f$  from (8) yields the vector

$$W = [ X \quad \dots \quad X^{10} \quad X^{10}Y \quad \dots \quad X^{10}Y^{20} \quad X^{10}Y^{20}X \quad \dots \quad X^{10}Y^{20}X^{10} ]^t$$

of length 40. Problems of this size are easily handled by today's SDP solvers. Nevertheless we provide a strengthening of our Newton chip algorithm reducing the number of words needed further (see §4.2), in this example to 2.

**Remark 3.3.** In Algorithm 2 the set of admissible words  $\mathcal{D}$  can be reduced further by using a common generalization of the total degree and the  $i$ -degree.

Consider the  $v$ -degree  $\deg_v$  of a monomial or polynomial in  $\mathbb{R}\langle\bar{X}\rangle$ , where  $v \in \mathbb{R}_{\geq 0}^n$  is a vector of nonnegative real “weights” and the  $v$ -degree of a monomial  $w \in \langle\bar{X}\rangle$  is the standard scalar product between  $v$  and the exponent of the commutative representative of  $w$ , i.e., for  $w = X_{i_1}^{e_1} \dots X_{i_r}^{e_r}$  the  $v$ -degree is  $\sum_{j=1}^r e_j v_{i_j}$ . This extends naturally to the  $v$ -degree and min- $v$ -degree of a polynomial  $f \in \mathbb{R}\langle\bar{X}\rangle$ . (The total degree corresponds to the  $v$  with all ones and the individual  $i$ -degrees correspond to the standard unit vectors.)

Now  $\mathcal{D}$  in Algorithm 2 can be replaced by

$$\mathcal{D} := \{w \in \langle\bar{X}\rangle \mid \forall v \in \mathbb{R}_{\geq 0}^n : \frac{\min \deg_v f}{2} \leq \deg_v w \leq \frac{\deg_v f}{2}\}$$

and the proof of Theorem 3.1 works verbatim in this new setting.

## 4. IMPLEMENTATION

**4.1. SDP duality.** An SDP of the form (PSDP) is said to satisfy *strong duality* if the optimal values of (PSDP) and (DSDP) are the same. Note: this includes the case where the primal (dual) problem is infeasible and the dual (primal) is unbounded. We refer the reader to [Tod01] for more on duality properties of SDP.

A sufficient condition for strong duality is the existence of a strictly feasible solution for at least one of the primal-dual pair of semidefinite programs, i.e., there exists a positive *definite* matrix  $G$  with  $\langle A_i, G \rangle = b_i$  for all  $i$  or there exists a vector  $y$  such that  $Z = C - \sum_i y_i A_i$  is positive *definite*. In this case there is no duality gap, that is, the optimal values of the primal and the dual are the same. Existence of a strictly feasible solution is also known as the *Slater condition*.

If the primal semidefinite program has a strictly feasible solution and the dual semidefinite problem is feasible, then both optimal values are finite and the optimal value of the dual is attained. An analogous statement holds if the dual has a strictly feasible solution.

As the following example demonstrates, the Slater condition is not necessarily satisfied on the primal side in our class of (SOHS<sub>SDP</sub>) problems.

**Example 4.1.** Let  $f = (XY + X^2)^*(XY + X^2)$ . It is homogeneous, hence there exists a unique self-adjoint Gram matrix

$$G = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

for  $f$  such that

$$f = \begin{bmatrix} YX & X^2 \end{bmatrix} G \begin{bmatrix} YX & X^2 \end{bmatrix}^*.$$

Clearly  $G$ , a rank 1 matrix, is the only feasible solution of  $(\text{SOHS}_{\text{SDP}})$ , hence the corresponding SDP has no strictly feasible solution on the primal side.

Nevertheless, since the objective function in our primal SDP is  $\langle I, G \rangle$ , the pair  $y = 0$ ,  $Z = I$  is always strictly feasible for the dual problem of  $(\text{SOHS}_{\text{SDP}})$  and thus we *do* have the strong duality property.

Hence, when the given NC polynomial is in  $\Sigma^2$ , the corresponding semidefinite program  $(\text{SOHS}_{\text{SDP}})$  is feasible and the optimal value is attained. If there is no strictly feasible solution then numerical difficulties might arise but state-of-the-art SDP solvers are able to overcome them in most of the instances. When the given NC polynomial is not in  $\Sigma^2$ , then the semidefinite problem  $(\text{SOHS}_{\text{SDP}})$  is infeasible and this might cause numerical problems as well. However, state-of-the-art SDP solvers (such as SeDuMi [SeD09] or SDPT3 [TTT09]) are robust and can reliably detect infeasibility for most practical problems. For more details see [dKRT98, PT07].

**4.2. Augmented Newton chip algorithm.** The following simple observation is often crucial in reducing the size of  $W$  returned by the Newton chip method.

**Lemma 4.2.** *If there exists a constraint of the form*

$$\langle A_w, G \rangle = 0$$

*in  $(\text{SOHS}_{\text{SDP}})$  and  $A_w$  is a diagonal matrix (i.e.,  $(A_w)_{u,u} = 2$  for some  $u \in W$  and  $A_w$  is 0 elsewhere), then we can eliminate  $u$  from  $W$  and update the  $(\text{SOHS}_{\text{SDP}})$ .*

*Proof.* Indeed, such a constraint implies that  $G_{u,u} = 0$  for the given  $u \in W$ , hence the  $u$ -th row and column of  $G$  must be zero, since  $G$  is positive semidefinite. So we can decrease the order of  $(\text{SOHS}_{\text{SDP}})$  by deleting the  $u$ -th row and column from  $G$ . ■

Lemma 4.2 applies if and only if there exists a constraint  $\langle A_w, G \rangle = 0$ , where  $w = u^*u$  for some  $u \in W$  and  $w \neq v^*z$  for all  $v, z \in W$ ,  $v \neq z$ . Therefore we augment the Newton chip method as shown in Algorithm 3.

Note that in Step 2 there might exist some word  $u \in W$  which does not satisfy the condition initially but after deleting another  $u'$  from  $W$  it does. We demonstrate Algorithm 3 in the following example.

INPUT:  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  with  $\deg f \leq 2d$ ,  $f = \sum_{w \in \langle \bar{X} \rangle} a_w w$ , where  $a_w \in \mathbb{R}$ .  
 STEP 1: Compute  $W$  by the Newton chip method (Algorithm 2).  
 STEP 2: While there exists  $u$  such that  $a_{u^*u} = 0$  and  $u^*u \neq v^*z$  for every pair  $v, z \in W$ ,  $v \neq z$ : delete  $u$  from  $W$ .  
 OUTPUT:  $W$ .

Algorithm 3: The augmented Newton chip method

**Example 4.3** (Example 2.5 continued). By applying the augmented Newton chip method to  $f$  from (8) we reduce the vector  $W$  significantly. Note that after Step 1 also the following words are in  $W$ :  $X^8$ ,  $X^9$ ,  $X^{10}$ . Although  $X^{18}$  does not appear in  $f$ , we cannot delete  $X^9$  from  $W$  immediately since  $X^{18} = (X^9)^*X^9 = (X^8)^*X^{10}$ . But we can delete  $X^{10}$  since  $X^{20}$  also does not appear in  $f$  and  $(X^{10})^*X^{10}$  is the unique decomposition of  $X^{20}$  inside  $W$ . After deleting  $X^{10}$  from  $W$  we realize that  $(X^9)^*X^9$  becomes the unique decomposition of  $X^{18}$ , hence we can eliminate  $X^9$  too. Eventually the augmented Newton chip method returns

$$W = [ X \quad X^{10}Y^{20}X^{10} ]^t,$$

which is exactly the minimum vector needed for the SOHS decomposition of  $f$ .

## 5. AN APPLICATION: OPTIMIZATION OF NC POLYNOMIALS

We implemented the Gram matrix method together with the augmented Newton chip method in the MATLAB software package named `NCsOSTools` which will be presented in detail elsewhere (see [CKP09]) and is freely available at

<http://ncsostools.fis.unm.si/downloads/>.

One of the main features of this package is `NCsos` which finds a SOHS decomposition of a given polynomial, if one exists, or returns that none exists.

In this section we present a “practical” application of SOHS decompositions, namely finding global minima of NC polynomials. (Readers interested in solving sums of squares problems for commuting polynomials are referred to one of the great existing packages `SOSTOOLS` [SOS09, PPSP05], `GloptiPoly` [HLL09], `YALMIP` [YAL09, Löf04], and `SparsePOP` [WKKM06].)

Unlike optimization of polynomials in commuting variables that requires a *sequence* of SDPs to compute the minimum, for NC polynomials a single SDP suffices by Helton’s theorem [Hel02]:  $f(A_1, \dots, A_n) \succeq aI$  for  $a \in \mathbb{R}$  and for all self-adjoint matrices  $A_i$  of the same size if and only if  $f - a \in \Sigma^2$ . The largest such  $a$  is obtained by solving the SDP

$$\begin{aligned} & \sup \quad a \\ & \text{s. t.} \quad f - a \in \Sigma^2. \end{aligned} \tag{SDP}_{\min}$$

Suppose  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle$  is of degree  $\leq 2d$ . Let  $W$  be a vector consisting of all monomials from  $\langle \bar{X} \rangle$  needed in the SOHS decomposition, i.e.,  $W$  is obtained by the (augmented) Newton chip method if  $f$  has nonzero constant term, otherwise we modify the (augmented) Newton chip method by setting  $m_i = m = 0$  in Step 3 of Algorithm 2). Assume the first entry of  $W$  is 1. Then  $(\text{SDP}_{\min})$  rewrites into

$$\begin{aligned} & \sup f_1 - \langle E_{11}, F \rangle \\ \text{s. t.} \quad & f - f_1 = W^*(F - F_{11}E_{11})W \\ & F \succeq 0. \end{aligned} \quad (\text{SDP}_{\min'})$$

(Here  $f_1$  is the constant term of  $f$  and  $E_{11}$  is the matrix with all entries 0 except for the  $(1, 1)$  entry which is 1.)

In general  $(\text{SDP}_{\min})$  does not satisfy the Slater condition. Nevertheless:

**Theorem 5.1.**  $(\text{SDP}_{\min})$  satisfies strong duality.

*Proof.* Suppose  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle \setminus \mathbb{R}$  is of degree  $\leq 2d$  and bounded from below. In particular, this implies that the highest homogeneous part of  $f$  is a sum of hermitian squares. Let  $\Sigma_{\leq 2d}^2$  denote the cone of all sums of hermitian squares of degree  $\leq 2d$ , i.e.,

$$\Sigma_{\leq 2d}^2 = \left\{ \sum_{i=1}^t g_i^* g_i \mid t \in \mathbb{N}, g_i \in \mathbb{R}\langle \bar{X} \rangle \text{ of degree } \leq d \right\}.$$

Then  $(\text{SDP}_{\min})$  can be rewritten as:

$$\begin{aligned} & \sup a \\ \text{s. t.} \quad & f - a \in \Sigma_{\leq 2d}^2. \end{aligned} \quad (\text{Primal})$$

The dual cone of  $\Sigma_{\leq 2d}^2$  is the set of all linear maps  $\text{Sym } \mathbb{R}\langle \bar{X} \rangle_{\leq 2d} \rightarrow \mathbb{R}$  which are nonnegative on  $\Sigma_{\leq 2d}^2$ . (We use  $(\text{Sym})\mathbb{R}\langle \bar{X} \rangle_{\leq 2d}$  to denote the set of all (symmetric) NC polynomials of degree  $\leq 2d$ .)

FACT: The cone  $\Sigma_{\leq 2d}^2$  is closed in  $\text{Sym } \mathbb{R}\langle \bar{X} \rangle_{\leq 2d}$ .

*Proof:* This is a well-known variant of the analogous claim in the commutative setting. See e.g. [MP05, Proposition 3.4] for a proof.

Let us now return to the SDP. To construct the dual problem to (Primal), we proceed as follows. (Primal) and its equivalent form  $(\text{SDP}_{\min'})$  can be with the help of the matrices  $A_w$  introduced below equation (7) on page 7, written as:

$$\begin{aligned} & \sup f_1 - \frac{1}{2} \langle A_1, F \rangle \\ \text{s. t.} \quad & f_w + f_w^* = \langle A_w, F \rangle \quad \forall w \in W_{2d} \setminus \{1\} \\ & F \succeq 0. \end{aligned} \quad (\text{Primal}')$$

Here  $W_{2d}$  is the set of all words of degree  $\leq 2d$  and  $2E_{11} = A_1$ . Hence the dual SDP to (Primal') is

$$\begin{aligned} \inf \quad & f_1 + \sum_{w \in U_{2d} \setminus \{1\}} y_w (f_w + f_{w^*}) \\ \text{s. t.} \quad & \frac{1}{2}A_1 + \sum_{w \in U_{2d} \setminus \{1\}} y_w A_w \succeq 0, \end{aligned} \tag{Dual'}$$

where  $U_{2d}$  is a set consisting of one words out of each pair  $(w, w^*)$  with  $w \in W_{2d}$ . Each feasible vector  $(y_w)_w$  of (Dual') gives rise to a linear map  $L : \mathbb{R}\langle \bar{X} \rangle_{\leq 2d} \rightarrow \mathbb{R}$  with  $L(1) = 1$  and  $L(w)$  equals  $y_w$  or  $y_{w^*}$  otherwise. This  $L$  satisfies  $L(1) = 1$ ,  $L = L \circ *$  and  $L(\Sigma_{\leq 2d}^2) \subseteq [0, \infty)$ .

Conversely, every such  $L$  yield a vector  $y_w = L(w)$ ,  $w \in W_{2d}$ . To see why the positivity constraint in (Dual') holds, note that  $L$  induces a map  $\text{Sym } \mathbb{R}^{N \times N} \rightarrow \mathbb{R}$  by

$$B \mapsto \langle B, \frac{1}{2}A_1 + \sum_{w \in U_{2d} \setminus \{1\}} y_w A_w \rangle = L(W_d^* B W_d).$$

By the positivity assumption on  $L$ , this map sends positive semidefinite matrices to nonnegative reals. Hence by the self-duality of the cone of all positive semidefinite matrices,  $\frac{1}{2}A_1 + \sum_{w \in U_{2d} \setminus \{1\}} y_w A_w \succeq 0$ .

Combining these observations we can present (Dual') as

$$\begin{aligned} \inf \quad & L(f) \\ \text{s. t.} \quad & L : \text{Sym } \mathbb{R}\langle \bar{X} \rangle_{\leq 2d} \rightarrow \mathbb{R} \quad \text{is linear} \\ & L(1) = 1 \\ & L(p^*p) \geq 0 \quad \text{for all } p \in \mathbb{R}\langle \bar{X} \rangle_{\leq d}. \end{aligned} \tag{Dual}$$

Let  $\underline{f}$  and  $\bar{f}$  denote the optimal value of (Primal) and (Dual), respectively. We claim that  $\underline{f} = \bar{f}$ . Clearly,  $\underline{f} \leq \bar{f}$ . (Dual) is always feasible (e.g.  $L$  mapping each polynomial into its constant term is feasible), hence  $\bar{f} < \infty$ .

Suppose first that (Primal) is feasible, hence  $\bar{f} \geq \underline{f} > -\infty$ . Note that  $L(f - \bar{f}) \geq 0$  for all  $L$  in the dual cone of  $\Sigma_{\leq 2d}^2$ . This means that  $f - \bar{f}$  belongs to the closure of  $\Sigma_{\leq 2d}^2$ , so by the Fact,  $f - \bar{f} \in \Sigma_{\leq 2d}^2$ . Hence also  $\underline{f} \geq \bar{f}$ .

Let us consider the case when (Primal) is infeasible, i.e.,  $f \in \text{Sym } \mathbb{R}\langle \bar{X} \rangle_{\leq 2d}$  is not bounded from below. Then for every  $a \in \mathbb{R}$ ,  $f - a$  is not an element of the closed convex cone  $\Sigma_{\leq 2d}^2$ . Thus by the Hahn-Banach separation theorem, there exists  $L : \text{Sym } \mathbb{R}\langle \bar{X} \rangle_{\leq 2d} \rightarrow \mathbb{R}$  satisfying  $L(\Sigma_{\leq 2d}^2) \subseteq [0, \infty)$ ,  $L(1) = 1$  and  $L(f) < a$ . As  $a$  was arbitrary, this shows that (Dual) is unbounded, hence strong duality holds in this case as well.  $\blacksquare$

Optimization of NC polynomials is implemented in our software package `NC-SOSTools`, where the optimal solution of (Primal) is computed by calling the routine `NCmin`.

**Acknowledgments.** We thank Tamás Terlaky and Markus Schweighofer for sharing their expertise on semidefinite programming and Kristijan Cafuta for diligent reading and his help with the implementation. The first author also Bill Helton for helpful discussions. We thank the anonymous referee for providing many helpful comments and suggestions.

## REFERENCES

- [BPR06] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, available from <http://perso.univ-rennes1.fr/marie-francoise.roy/bpr-posted1.html>, 2006.
- [BTN01] A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization*. MPS/SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001. Analysis, algorithms, and engineering applications.
- [CKP09] K. Cafuta, I. Klep, and J. Povh. NCSOSTools: a computer algebra system for symbolic and numerical computation with noncommutative polynomials, available from [http://ncsostools.fis.unm.si/downloads/ncsostools\\_manual.pdf](http://ncsostools.fis.unm.si/downloads/ncsostools_manual.pdf). 6 Mar 2009.
- [CLR95] M.D. Choi, T.Y. Lam, and B. Reznick. Sums of squares of real polynomials. In *K-theory and algebraic geometry: connections with quadratic forms and division algebras (Santa Barbara, CA, 1992)*, volume 58 of *Proc. Sympos. Pure Math.*, pages 103–126. Amer. Math. Soc., Providence, RI, 1995.
- [dKRT98] E. de Klerk, C. Roos, and T. Terlaky. Infeasible-start semidefinite programming algorithms via self-dual embeddings. In *Topics in semidefinite and interior-point methods (Toronto, ON, 1996)*, volume 18 of *Fields Inst. Commun.*, pages 215–236. Amer. Math. Soc., Providence, RI, 1998.
- [dOHMP08] M.C. de Oliveira, J.W. Helton, S. McCullough, and M. Putinar. Engineering systems and free semi-algebraic geometry. In *Emerging Applications of Algebraic Geometry*, volume 149 of *IMA Vol. Math. Appl.*, pages 17–62. Springer, 2008.
- [Hel02] J.W. Helton. “Positive” noncommutative polynomials are sums of squares. *Ann. of Math. (2)*, 156(2):675–694, 2002.
- [HLL09] D. Henrion, J.-B. Lasserre, and J. Löfberg. GloptiPoly 3: moments, optimization and semidefinite programming, available from <http://www.laas.fr/~henrion/software/gloptipoly3/>. 1 Mar 2009.
- [KS08a] I. Klep and M. Schweighofer. Connes’ embedding conjecture and sums of Hermitian squares. *Adv. Math.*, 217(4):1816–1837, 2008.
- [KS08b] I. Klep and M. Schweighofer. Sums of Hermitian squares and the BMV conjecture. *J. Stat. Phys.*, 133(4):739–760, 2008.
- [Löf04] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.
- [Mit09] H. Mittelmann. <http://plato.asu.edu/sub/pns.html>. 1 Mar 2009.
- [MP05] S. McCullough and M. Putinar. Noncommutative sums of squares. *Pacific J. Math.*, 218(1):167–171, 2005.

- [MPRW09] J. Malick, J. Povh, F. Rendl, and A. Wiegele. Regularization methods for semidefinite programming. *SIAM Journal on Optimization*, 20(1):336–356, 2009.
- [Par03] P.A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Math. Program.*, 96(2, Ser. B):293–320, 2003.
- [PK97] L. Porkolab and L. Khachiyan. On the complexity of semidefinite programs. *J. of Global Optimization*, 10(4):351–365, 1997.
- [PPSP05] S. Prajna, A. Papachristodoulou, P. Seiler, and P.A. Parrilo. SOSTOOLS and its control applications. In *Positive polynomials in control*, volume 312 of *Lecture Notes in Control and Inform. Sci.*, pages 273–292. Springer, Berlin, 2005.
- [PRW06] J. Povh, F. Rendl, and A. Wiegele. A boundary point method to solve semidefinite programs. *Computing*, 78:277–286, 2006.
- [PT07] I. Pólik and T. Terlaky. New stopping criteria for detecting infeasibility in conic optimization. *AdvOL-report #2007/09*, 2007.
- [PW98] V. Powers and T. Wörmann. An algorithm for sums of squares of real polynomials. *J. Pure Appl. Algebra*, 127(1):99–104, 1998.
- [Ram97] M.V. Ramana. An exact duality theory for semidefinite programming and its complexity implications. *Math. Programming*, 77(2, Ser. B):129–162, 1997.
- [Rez78] B. Reznick. Extremal PSD forms with few terms. *Duke Math. J.*, 45(2):363–374, 1978.
- [RFP07] B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization, available from <http://arxiv.org/abs/0706.4138>. 2007.
- [Row80] L.H. Rowen. *Polynomial identities in ring theory*, volume 84 of *Pure and Applied Mathematics*. Academic Press Inc., New York, 1980.
- [SeD09] SeDuMi. <http://sedumi.ie.lehigh.edu/>. 1 Mar 2009.
- [SOS09] SOSTools. <http://www.cds.caltech.edu/sostools/>. 1 Mar 2009.
- [Tod01] M.J. Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, 2001.
- [TTT09] K.-C. Toh, M.J. Todd, and R.-H. Tütüncü. SDPT3 version 4.0 (beta) – a MATLAB software for semidefinite-quadratic-linear programming, available from <http://www.math.nus.edu.sg/~mattokhc/sdpt3.html>. 1 Mar 2009.
- [WKKM06] H. Waki, S. Kim, M. Kojima, and M. Muramatsu. Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity. *SIAM J. Optim.*, 17(1):218–242 (electronic), 2006.
- [WSV00] H. Wolkowicz, R. Saigal, and L. Vandenberghe. *Handbook of Semidefinite Programming*. Kluwer, 2000.
- [YAL09] YALMIP. <http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php>. 1 Mar 2009.

IGOR KLEP, UNIVERZA V LJUBLJANI, FAKULTETA ZA MATEMATIKO IN FIZIKO, JADRANSKA 19, 1111 LJUBLJANA, AND UNIVERZA V MARIBORU, FAKULTETA ZA NARAVOSLOVJE IN MATEMATIKO, KOROŠKA 160, 2000 MARIBOR, SLOVENIA

*E-mail address:* igor.klep@fmf.uni-lj.si

JANEZ POVH, INSTITUTE OF MATHEMATICS, PHYSICS AND MECHANICS LJUBLJANA, JADRANSKA 19, 1111 LJUBLJANA, SLOVENIA

*E-mail address:* janez.povh@fis.unm.si

## NOT FOR PUBLICATION

## CONTENTS

1. Introduction	1
1.1. Notation	1
1.2. Motivation and Contribution	2
2. Sums of hermitian squares	3
2.1. Sums of hermitian squares and Gram matrices	3
2.2. Semidefinite programming	5
2.3. Sums of hermitian squares and SDP	7
3. Newton chip method	9
4. Implementation	11
4.1. SDP duality	11
4.2. Augmented Newton chip algorithm	12
5. An application: Optimization of NC polynomials	13
Acknowledgments.	16
References	16