A bounded degree SOS hierarchy for large scale polynomial optimization with sparsity

Tillmann Weisser, Jean-Bernard Lasserre, Kim-Chuan Toh

A bounded degree SOS hierarchy for large scale polynomial optimization with sparsity *

T. Weisser† Jean B. Lasserre‡ and Kim-Chuan Toh§

July 5, 2016

Abstract

We provide a sparse version of the bounded degree SOS (BSOS) hierarchy for polynomial optimization problems. The presented version permits to handle large scale problems which satisfy a structured sparsity pattern. When the sparsity pattern satisfies the running intersection property, this sparse BSOS hierarchy of semidefinite programs (with semidefinite constraints of fixed size) converges to the global optimum of the original problem. Moreover, for the class of SOS-convex problems, finite convergence takes place at the first step of the hierarchy, just as in the dense version.

Keywords: Global Optimization, Semidefinite Programming, Sparsity, Large Scale Problems, Convex Relaxations, Positivity Certificates
MSC: 90C26, 90C22

1 Introduction

We consider the polynomial optimization problem:

$$(P) \quad f^* := \min_{\mathbf{x}} \{ f(\mathbf{x}) \ : \ \mathbf{x} \in \mathbf{K} \} \tag{1}$$

where $f \in \mathbb{R}[\mathbf{x}]$ is a polynomial and $\mathbf{K} \subset \mathbb{R}^n$ is the basic semi-algebraic set

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^n : g_j(\mathbf{x}) \geq 0, j = 1, \ldots, m\}, \tag{2}$$

for some polynomials $g_j \in \mathbb{R}[\mathbf{x}]$, $j = 1, \ldots, m$. In [14] we provided a new hierarchy of semidefinite programs $(\boldsymbol{Q}_d^k)$ indexed by $d \in \mathbb{N}$ and parametrized by $k \in \mathbb{N}$ (fixed), whose associated (monotone non decreasing) sequence of optimal values $(\rho_d^k)_{d \in \mathbb{N}}$, converges to f^* as $d \to \infty$, i.e., $\rho_d^k \to f^*$ as $d \to \infty$.

One important distinguishing feature of the BSOS hierarchy (when compared to the standard SOS hierarchy defined in [15, 20]) is that for each semidefinite relaxation $\boldsymbol{Q}_d^k$, $d \in \mathbb{N}$, the size of the semidefinite constraint is $O(n^k)$, hence fixed and controlled by the parameter k (fixed and chosen by the user). With $k = 0$ one retrieves the LP-hierarchy based on a positivity certificate due to Stengle; see [14] and [16] for more details.

*The work of the first author is partially supported by a PGMO grant from *Fondation Mathématique Jacques Hadamard* and a grant from the *ERC council* for the Taming project (ERC-Advanced Grant #666981 TAMING).

†LAAS-CNRS, University of Toulouse, LAAS, 31031 Toulouse cédex 4, France (`tweisser@laas.fr`).

‡LAAS-CNRS and Institute of Mathematics, University of Toulouse, LAAS, 31031 Toulouse cédex 4, France (`lasserre@laas.fr`).

§Department of Mathematics, National University of Singapore, 10 Lower Kent Ridge Road, Singapore 119076 (`mattohkc@nus.edu.sg`).

Another distinguishing feature of the BSOS hierarchy (when compared to the LP-hierarchy defined in [16]) is *finite convergence* for an important class of convex problems. That is, when $f, -g_j$ are SOS-convex polynomials of degree bounded by $2k$, then the first semidefinite relaxation of the hierarchy $(\boldsymbol{Q}_d^k)_{d\in\mathbb{N}}$, is exact, i.e., $\rho_1^k = f^*$. (In contrast the LP-hierarchy cannot converges in finitely many steps for such convex problems).

Contribution

Even though the size $O(n^k)$ of the semidefinite constraint of $\boldsymbol{Q}_d^k$ is fixed for all d and permits to handle problems (P) of size larger than with the standard SOS-hierarchy, it still limits the application of the BSOS hierarchy to problems of relatively modest size (say medium size problems). The present contribution is to provide a *sparse* version of the BSOS hierarchy which permits to handle large size problems (P) that satisfy some (structured) sparsity pattern. The sparse BSOS hierarchy is the analogue for the BSOS hierarchy of the sparse version for the standard SOS-hierarchy introduced by Waki et al. [33]. Again as in the dense case, a distinguishing feature of the former (and in contrast to the latter) is that the size of the resulting semidefinite constraints is *fixed in advance* at the user convenience and does *not* depend on the rank in the hierarchy. However, such an extension is not straightforward because in contrast to Putinar's SOS-based certificate (where the g_j's appear separately), the positivity certificate used in the dense BSOS algorithm [14] potentially mixes all polynomials g_j that define $\mathbf{K}$, that is, if f is positive on $\mathbf{K}$ then

$$f \;=\; \sigma + \sum_{\alpha,\beta\in\mathbb{N}^m} c_{\alpha\beta} \prod_{j=1}^{m} g_j^{\alpha_j}(1-g_j)^{\beta_j}, \tag{3}$$

for some SOS polynomial σ and positive scalar weights $c_{\alpha\beta}$. Therefore in principle the sparsity as defined in [33] may be destroyed in σ and in the products $\prod_j g_j^{\alpha_j}(1-g_j)^{\beta_j}$. In fact, one contribution of this paper is to provide a specialized sparse version of (3). In particular, we prove that if the sparsity pattern satisfies the so-called *Running Intersection Property* (RIP) then the sparse-BSOS hierarchy also converges to the global optimum f^* of (P). A sufficient rank-condition also permits to detect finite convergence. At last but not least, we also prove that the sparse BSOS hierarchy preserves a distinguishing feature of the dense BSOS hierarchy, namely its finite convergence at the first step of the hierarchy for the class of SOS-convex problems. (Recall that the standard LP hierarchy cannot converge in finitely many steps for such problems [16, 21].)

Roughly speaking we say that (P) in (1) satisfies a structured sparsity pattern, if the set $I_0 := \{1,\ldots,n\}$ of all variables is some union $\cup_{k=1}^p I_k$ of smaller blocks of variables I_k such that each monomial of the objective function only consists of variables in one of the blocks. In addition, each polynomial g_j in the definition (2) of the feasible set, is also a polynomial only in variables of one of the blocks. Of course the blocks (I_k) may overlap, i.e., variables may appear in several blocks. Together with the maximum degree appearing in the data of (P), the number and size of the blocks (I_k) as well as the size of their overlaps, are the characteristics of the sparsity pattern which have the strongest influence on the performance of our algorithm.

Computational experiments. We have tested the sparse BSOS algorithm on a variety of small and large scale examples with different degrees in the data. In a comparison to the dense BSOS version on this sample (when the size of (P) permits), the sparse approach behaves very well as the global minimum is attained whenever the dense version succeeded. Some medium size problems with non convex quadratic objective functions randomly generated, illustrate how the block and overlap sizes influence the behavior of the algorithm. Next, we have tested large-scale quadratic examples (also randomly generated). Problems with up to 3000 variables can be solved

in about 200 seconds on a lap-top[1]. Finally we have tested our sparse BSOS implementation on some typical problems from the literature in non linear optimization.

From this first set of examples it seems that the sparse version of the BSOS algorithm is able to solve non convex problems of significant size ($n \approx 1000$ for quartic problems and $n \approx 3000$ for quadratic problems) provided that the maximum size of the blocks is relatively modest (say at most 4 for quartic problems, less than 10 for quadratic problems).

2 Preliminaries

2.1 Notation and definitions

Let $\mathbb{R}[\mathbf{x}]$ be the ring of polynomials in the variables $\mathbf{x} = (x_1, \ldots, x_n)$. Denote by $\mathbb{R}[\mathbf{x}]_d \subset \mathbb{R}[\mathbf{x}]$ the vector space of polynomials of degree at most d, which has dimension $s(d) := \binom{n+d}{d}$, with e.g., the usual canonical basis $(\mathbf{x}^\gamma)_{\gamma \in \mathbb{N}^n_d}$ of monomials, where $\mathbb{N}^n_d := \{\gamma \in \mathbb{N}^n : |\gamma| \leq d\}$. Also, denote by $\Sigma[\mathbf{x}] \subset \mathbb{R}[\mathbf{x}]$ (resp. $\Sigma[\mathbf{x}]_d \subset \mathbb{R}[\mathbf{x}]_{2d}$) the cone of sums of squares (s.o.s.) polynomials (resp. s.o.s. polynomials of degree at most $2d$). If $f \in \mathbb{R}[\mathbf{x}]_d$, we write $f(\mathbf{x}) = \sum_{\gamma \in \mathbb{N}^n_d} \mathrm{f}_\gamma \mathbf{x}^\gamma$ in the canonical basis and denote by $\mathbf{f} = (\mathrm{f}_\gamma)_\gamma \in \mathbb{R}^{s(d)}$ its vector of coefficients. Finally, let $\mathcal{S}^n$ denote the space of $n \times n$ real symmetric matrices, with inner product $\langle \mathbf{A}, \mathbf{B} \rangle = \operatorname{trace} \mathbf{AB}$. We use the notation $\mathbf{A} \succeq 0$ (resp. $\mathbf{A} \succ 0$) to denote that $\mathbf{A}$ is positive semidefinite (definite). With $g_0 := 1$, the quadratic module $Q(g_1, \ldots, g_m) \subset \mathbb{R}[\mathbf{x}]$ generated by polynomials $g_1, \ldots, g_m$, is defined by

$$Q(g_1, \ldots, g_m) := \left\{ \sum_{j=0}^{m} \sigma_j \, g_j \ : \ \sigma_j \in \Sigma[\mathbf{x}] \right\}.$$

With a real sequence $\boldsymbol{y} = (y_\gamma)_{\gamma \in \mathbb{N}^n_d}$, one may associate the linear functional $L_{\boldsymbol{y}} : \mathbb{R}[\mathbf{x}]_d \to \mathbb{R}$ defined by

$$f \left(= \sum_\gamma \mathrm{f}_\gamma \, \mathbf{x}^\gamma \right) \quad \mapsto L_{\boldsymbol{y}}(f) := \sum_\gamma \mathrm{f}_\gamma \, y_\gamma,$$

which is called the Riesz functional. If $d = 2a$ denote by $\mathbf{M}_a(\boldsymbol{y})$ the moment matrix associated with $\boldsymbol{y}$. It is a real symmetric matrix with rows and columns indexed in the basis of monomials $(\mathbf{x}^\gamma)_{\gamma \in \mathbb{N}^n_a}$, and with entries

$$\mathbf{M}_a(\boldsymbol{y})(\alpha, \beta) := L_{\boldsymbol{y}}(\mathbf{x}^{\alpha+\beta}) = y_{\alpha+\beta}, \qquad \forall \, \alpha, \beta \in \mathbb{N}^n_a.$$

If $\boldsymbol{y} = (y_\gamma)_{\gamma \in \mathbb{N}^n}$ is the sequence of moments of some Borel measure μ on $\mathbb{R}^n$ then $\mathbf{M}_a(\boldsymbol{y}) \succeq 0$ for all $a \in \mathbb{N}$. However the converse is not true in general.

A polynomial $f \in \mathbb{R}[\mathbf{x}]$ is said to be SOS-convex if its Hessian matrix $\mathbf{x} \mapsto \nabla^2 f(\mathbf{x})$ is an SOS matrix-polynomial, that is, $\nabla^2 f = \boldsymbol{L}\,\boldsymbol{L}^T$ for some real matrix polynomial $\boldsymbol{L} \in \mathbb{R}[\mathbf{x}]^{n \times a}$ (for some integer a). In particular, for SOS-convex polynomials and sequences $\boldsymbol{y}$ with positive semidefinite moment matrix $\mathbf{M}_a(\boldsymbol{y}) \succeq 0$, a Jensen-type inequality is valid:

Lemma 1. *Let $f \in \mathbb{R}[\mathbf{x}]_{2a}$ be SOS-convex and let $\boldsymbol{y} = (y_\gamma)_{\gamma \in \mathbb{N}^n_{2a}}$ be such that $y_0 = 1$ and $\mathbf{M}_a(\boldsymbol{y}) \succeq 0$. Then*

$$L_{\boldsymbol{y}}(f) \geq f(L_{\boldsymbol{y}}(\mathbf{x})), \quad \textit{with} \quad L_{\boldsymbol{y}}(\mathbf{x}) := (L_{\boldsymbol{y}}(x_1), \ldots, L_{\boldsymbol{y}}(x_n)).$$

For a proof see Theorem 13.21, p. 209 in [21].

[1]The numerical experiments were run on a standard lap-top from the year 2015. For a more detailed description see page 9.

2.2 A sparsity pattern

Given $I \subset \{1,\dots,n\}$ denote by $\mathbb{R}[\mathbf{x}; I]$ the ring of polynomials in the variables $\{x_i : i \in I\}$, which we understand as a subring of $\mathbb{R}[\mathbf{x}]$. Hence, a polynomial $g \in \mathbb{R}[\mathbf{x}; I]$ canonically induces two polynomial functions $g : \mathbb{R}^{\#I} \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}$.

Assumption 1 (Sparsity pattern)**.** *There exists $p \in \mathbb{N}$ and subsets $I_\ell \subseteq \{1,\dots,n\}$ and $J_\ell \subseteq \{1,\dots,m\}$ for all $\ell \in \{1,\dots,p\}$ such that*

- *$f = \sum_{\ell=1}^p f^\ell$, with $f^\ell \in \mathbb{R}[\mathbf{x}; I_\ell]$ each,*
- *$g_j \in \mathbb{R}[\mathbf{x}; I_\ell]$ for all $j \in J_\ell$ and $\ell \in \{1,\dots,p\}$,*
- *$\bigcup_{\ell=1}^p I_\ell = \{1,\dots,n\}$,*
- *$\bigcup_{\ell=1}^p J_\ell = \{1,\dots,m\}$;*
- *for all $\ell = 1,\dots,p-1$ there is an $s \leq \ell$ such that $\left(I_{\ell+1} \cap \bigcup_{r=1}^{\ell} I_r\right) \subseteq I_s$ (Running Intersection Property).*

From now on, we assume that $\mathbf{K}$ is described by polynomials $g_1,\dots,g_m$ such that

$$\mathbf{K} = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq g_j(\mathbf{x}) \leq 1, \quad j = 1,\dots,m\}. \tag{4}$$

Note that this is no restriction to $\mathbf{K}$ defined in (1) if $\mathbf{K}$ is compact, as the constraint polynomials can be scaled by a positive factor without adding or losing information.

Let Assumption 1 hold, define $n_\ell := |I_\ell|$, $m_\ell := |J_\ell|$, and let $\mathbf{K}_\ell \subset \mathbb{R}^{n_\ell}$, $\ell = 1,\dots,p$, be the sets:

$$\mathbf{K}_\ell := \{\mathbf{x} \in \mathbb{R}^{n_\ell} : 0 \leq g_j(\mathbf{x}) \leq 1, \quad j \in J_\ell\}, \qquad \ell = 1,\dots,p. \tag{5}$$

Define $\pi_\ell : \mathbb{R}^n \to \mathbb{R}^{n_\ell}, \mathbf{x} \mapsto (x_i)_{i \in I_\ell}$. Then

$$K = \{\mathbf{x} \in \mathbb{R}^n : \pi_\ell(\mathbf{x}) \in K_\ell \text{ for all } \ell = 1,\dots,p\}. \tag{6}$$

Assumption 2. *For each ℓ there exists $M_\ell \geq 0$ and $j \in J_\ell$ such that $g_j = M_\ell - \sum_{i \in I_\ell} x_i^2$.*[2] *Furthermore, we assume that the polynomials 1 and $(g_j)_{j \in J_\ell}$ generate $\mathbb{R}[\mathbf{x}; I_\ell]$ as an $\mathbb{R}$-algebra for each $\ell = 1,\dots,p$.*

Note that the first part of Assumption 2 implies, that $\mathbf{K}$ is compact. On the other hand, if $\mathbf{K}$ is compact and Assumption 1 holds, Assumption 2 can easily be satisfied by adding the polynomials $M_\ell - \sum_{j \in J_\ell} x_j^2$ possibly after scaling.

2.3 A preliminary result

In this section we provide a sparse version of Stengle and Vasilescu Positivstellensatz. Let $N^\ell := \{(\alpha,\beta) \in \mathbb{N}^{2m} : \alpha_j = \beta_j = 0 \text{ if } j \notin J_\ell\}$.

Theorem 1 (Sparse Stengle Positivstellensatz)**.** *Let $f, g_1,\dots,g_m \in \mathbb{R}[\mathbf{x}]$ satisfy Assumption 1 and 2 . If f is (strictly) positive on $\mathbf{K}$ then $f = \sum_{\ell=1}^p f^\ell$ for some polynomials $f^\ell \in \mathbb{R}[\mathbf{x}; I_\ell]$, $\ell = 1,\dots,p$, and*

$$f^\ell = \sum_{(\alpha,\beta) \in N^\ell} c_{\alpha\beta}^\ell \prod_{j=1}^m g_j^{\alpha_j} (1 - g_j)^{\beta_j} \tag{7}$$

for finitely many positive weights $c_{\alpha\beta}^\ell$.

[2] In fact it is enough to assume, that a positive multiple of $M_\ell - \sum_{i \in I_\ell} x_i^2$ is contained in the description of $\mathbf{K}_\ell$ for each ℓ. We omit this in the assumption for a better readability.

Note that for each ℓ the representation (7) only involves $g_j \in \mathbb{R}[\mathbf{x}; I_\ell]$ since the corresponding exponents are 0 if $j \notin J_\ell$.

Proof. As f is positive on $\mathbf{K}$ there exist $\varepsilon > 0$ such that $f - \varepsilon > 0$ on $\mathbf{K}$. Under the assumptions of the theorem, by [18, Corollary 3.3] (a sparse version of Putinar's Positivstellensatz),

$$f - \varepsilon = \sum_{\ell=1}^{p} \left(\underbrace{\sigma_0^\ell + \sum_{j \in J_\ell} \sigma_j^\ell \, g_j}_{\in \mathbb{R}[\mathbf{x}; I_\ell] \text{ and } \geq 0 \text{ on } \mathbf{K}_\ell} \right),$$

for some SOS polynomials σ_j^ℓ. Next, let

$$f^\ell \; := \; \varepsilon/p + \sigma_0^\ell + \sum_{j \in J_\ell} \sigma_j^\ell \, g_j, \qquad \ell = 1, \ldots, p.$$

Notice that $f = \sum_{\ell=1}^{p} f^\ell$ and each f^ℓ is strictly positive on $\mathbf{K}_\ell$, $\ell = 1, \ldots, p$. As Assumption 2 holds, by [32, Theorem 1.3.1] (7) holds for each ℓ. □

3 Main result

3.1 The Sparse Bounded-SOS-hierarchy (Sparse-BSOS)

Consider problem (P) in (1) and let Assumption 1 and Assumption 2 hold. For $d \in \mathbb{N}$ let $N_d^\ell := \{(\alpha, \beta) \in \mathbb{N}^{2m} : \alpha_j = \beta_j = 0 \text{ if } j \notin J_\ell, \; \sum_j \alpha_j + \beta_j \leq d\}$ and for every $\ell = 1, \ldots, p$, let

$$h_{\alpha\beta}^\ell \; := \; \prod_{j=1}^{m} g_j^{\alpha_j} (1 - g_j)^{\beta_j} \; \in \; \mathbb{R}[\mathbf{x}; I_\ell], \quad (\alpha, \beta) \in N_d^\ell.$$

Let $k \in \mathbb{N}$ be fixed and define $d_{\max} := \max\{\deg(f), 2k, d \max_j\{\deg(g_j)\}$ consider the family of optimization problems indexed by $d \in \mathbb{N}$:

$$\begin{aligned} q_d^k := \sup \{ \, t \; : f^\ell - \sum_{(\alpha,\beta) \in N_d^\ell} \lambda_{\alpha\beta}^\ell h_{\alpha\beta}^\ell \in \Sigma[\mathbf{x}; I_\ell]_k, \quad \ell = 1, \ldots, p, \\ f \; - t \; = \; \sum_{\ell=1}^{p} f^\ell, \; \boldsymbol{\lambda}^\ell \geq 0, \; t \in \mathbb{R}, \; f^\ell \in \mathbb{R}[\mathbf{x}; I_\ell]_{d_{\max}} \Big\} . \end{aligned} \tag{8}$$

Observe that when k is fixed, then for each $d \in \mathbb{N}$, computing q_d^k in (8) reduces to solving a semidefinite program and hence, because $q_{d+1}^k \geq q_d^k$ for all $d \in \mathbb{N}$, (8) defines a hierarchy of semidefinite programs. To formulate these programs one has at least two possibilities depending on how the polynomial identities are implemented. To state that two polynomials $p, q \in \mathbb{R}[\mathbf{x}]_d$ are identical one can either *equate their coefficients* (e.g. in the monomial basis), i.e., $\mathrm{p}_\gamma = \mathrm{q}_\gamma$ for all $\gamma \in \mathbb{N}_d^n$, or one can *equate their values* on $\binom{n+d}{d}$ generic points (e.g. randomly generated on the box $[-1, 1]^n$). In contrast to the dense version [14] we have decided to use the former implementation. Equating coefficients is reasonable in the present context, since we assume the number of variables n_ℓ in each block to be rather small. The drawback of this choice is that going high in the relaxation order d is very time consuming. This effect becomes worse when the maximal degree of the g_j is high.

Define $\mathcal{I}_\ell^d := \{\gamma \in \mathbb{N}^n : \gamma_j = 0 \text{ if } j \notin I_\ell,\ |\gamma| \le d\}$. Then for k fixed and for each d, we get

$$\begin{aligned} q_d^k := \sup\{\, t \quad \text{s.t. } \mathrm{f}_\gamma^\ell - \sum_{(\alpha,\beta)\in N_d^\ell} \lambda_{\alpha\beta}^\ell\, (\mathrm{h}_{\alpha\beta}^\ell)_\gamma - \langle Q^\ell, \left(\boldsymbol{v}_k^\ell (\boldsymbol{v}_k^\ell)^T\right)_\gamma \rangle = 0, \quad \forall \gamma \in \mathcal{I}_\ell^{d_{\max}}, \\ \sum_{i:\,\gamma\in\mathcal{I}_i^{d_{\max}}} \mathrm{f}_\gamma^i = \mathrm{f}_\gamma - t\,1_{\gamma=0}, \quad \forall \gamma \in \bigcup_{j=1}^p \mathcal{I}_j^{d_{\max}}, \\ Q^\ell \in \mathcal{S}_+^{s(\ell,k)},\ \boldsymbol{\lambda}^\ell \ge 0,\ \mathbf{f}^\ell \in \mathbb{R}^{s(\ell,d_{\max})},\ \ell = 1,\dots,p,\ t \in \mathbb{R}\Big\}, \end{aligned} \tag{9}$$

where $s(\ell,k) := \binom{n_\ell+k}{k}$, and $\boldsymbol{v}_k^\ell$ is the vector of the canonical (monomial) basis of the vector space $\mathbb{R}[\mathbf{x}; I_\ell]_k$. Here we use the convention that the coefficient q_γ of a polynomial q is 0 if $|\gamma| > \deg(q)$. For a matrix polynomial $\mathbf{q} = (q_{ij})_{1\le i,j\le s} \in \mathbb{R}[\mathbf{x}]^{s\times s}$ the coefficient $\mathbf{q}_\gamma$ is the matrix $((q_{ij})_\gamma)_{1\le i,j\le s} \in \mathbb{R}^{s\times s}$. Note that the semidefinite matrix variables have fixed size $s(\ell,k)$, independent of $d \in \mathbb{N}$. This is a crucial feature for computational efficiency of the approach.

The dual of the semidefinite program (9) reads:

$$\begin{aligned} \tilde{q}_d^k := \inf\{\, L_{\boldsymbol{y}}(f) \quad \text{s.t. } \mathbf{M}_k(\boldsymbol{\theta}^\ell) \succeq 0; \quad L_{\boldsymbol{\theta}^\ell}(h_{\alpha\beta}^\ell) \ge 0, \quad (\alpha,\beta) \in N_d^\ell;\ \ell = 1,\dots,p \\ y_\gamma = \theta_\gamma^\ell, \quad \forall \ell : \gamma \in \mathcal{I}_\ell^{d_{\max}}; \quad \gamma \in \bigcup_{i=1}^p \mathcal{I}_i^{d_{\max}} \\ y_0 = 1\,\}. \end{aligned} \tag{10}$$

By standard weak duality of convex optimization, $\tilde{q}_d^k \ge q_d^k$ for all $d \in \mathbb{N}$. Moreover (10) is a relaxation of (P) in (1) and so $f^* \ge \tilde{q}_d^k \ge q_d^k$ for all $d \in \mathbb{N}$. In fact we even have the more precise and interesting result.

Theorem 2 ([17])**.** *Consider problem (P) in (1) and let Assumption 1 and Assumption 2 hold. Let $k \in \mathbb{N}$ be fixed. Then the sequence $(q_d^k)_{d\in\mathbb{N}}$, defined in (8) is monotone non-decreasing and $q_d^k \to f^*$ as $d \to \infty$.*

Proof. Monotonicity of the sequence $(q_d^k)_{d\in\mathbb{N}}$ follows from its definition. Let $\varepsilon > 0$ be fixed arbitrary. Then the polynomial $f - f^* + \varepsilon$ is positive on $\mathbf{K}$. By Theorem 1 there exist polynomials f^ℓ, $\ell = 1,\dots,p$, such that (7) holds, i.e.,

$$f - \underbrace{(f^* - \varepsilon)}_{t} = \sum_{\ell=1}^p f^\ell \quad \text{with} \quad f^\ell = \sum_{(\alpha,\beta)\in N^\ell} \underbrace{c_{\alpha\beta}^\ell}_{\ge 0}\, h_{\alpha\beta}^\ell, \quad \ell = 1,\dots,p,$$

for finitely many positive weights $c_{\alpha\beta}^\ell$. Hence $(f^* - \varepsilon, f^\ell, c_{\alpha\beta}^\ell)$ is a feasible solution for (8) as soon as d is sufficiently large, and therefore $q_d^k \ge f^* - \varepsilon$. Combining this with $q_d^k \le f^*$ and noting that $\varepsilon > 0$ was arbitrary, yield the desired result $q_d^k \to f^*$ as $d \to \infty$. □

We next show that a distinguishing feature of the dense BSOS hierarchy [14] is also valid for its sparse version.

Theorem 3. *Consider problem (P) in (1) and let Assumption 1 and Assumption 2 hold. Let $k \in \mathbb{N}$ be fixed and assume that for every $\ell = 1,\dots,p$, the polynomials f^ℓ and $-g_j$ are all SOS-convex polynomials of degree at most $2k$. (If $k > 1$ we assume (with no loss of generality) that for each $\ell = 1,\dots,p$, and some sufficiently large $\kappa_\ell > 0$, the redundant (SOS-convex) constraints $\kappa_\ell - \sum_{i\in I_\ell} x_i^{2k} \ge 0$, $\ell = 1,\dots,p$, are present in the description (4) of $\mathbf{K}$.)*

Then the semidefinite program (10) has an optimal solution $((\boldsymbol{\theta}^{*\ell}), \boldsymbol{y}^*)$ such that $f^* = \tilde{q}_1^k = L_{\boldsymbol{y}^*}(f)$ and $\mathbf{x}^* := (L_{\boldsymbol{y}^*}(x_1),\dots,L_{\boldsymbol{y}^*}(x_n)) \in \mathbf{K}$ is an optimal solution of (1). Hence finite convergence takes place at the first step of the hierarchy.

Proof. Let $d = 1$ (so that $d_{\max} = 2k$) and consider the semidefinite program (10). Note, that $\boldsymbol{\theta}^\ell = (\theta^\ell_\gamma)_{\gamma\in\mathcal{I}^{2k}_\ell}$. Recall that by Assumption 2, for every $\ell = 1,\ldots,p$, there exists $j \in J_\ell$ such that $g_j(\mathbf{x}) = M_\ell - \sum_{i\in I_\ell} x_i^2$. In addition if $k > 1$ then there also exists r such that $g_r(\mathbf{x}) = \kappa_\ell - \sum_{i\in I_\ell} x_i^{2k}$. Hence, feasibility in (10) (with an appropriate choice of $(\alpha,\beta) \in N^\ell_d$) implies that $L_{\boldsymbol{\theta}^{*\ell}}(g_j) \geq 0$ and $L_{\boldsymbol{\theta}^{*\ell}}(g_r) \geq 0$, which in turn imply:

$$L_{\boldsymbol{\theta}}(x_i^2) \leq M_\ell\,\theta^\ell_0\,(= M_\ell) \quad\text{and}\quad L_{\boldsymbol{\theta}}(x_i^{2k}) \leq \kappa_\ell\,\theta^\ell_0\,(=\kappa_\ell) \quad (\text{if } k>1), \quad \forall i\in I_\ell, \quad \forall \ell = 1,\ldots,p,$$

where we have used that $\theta^\ell_0 = y_0 = 1$ for all $\ell = 1,\ldots,p$.

Combining this with $\mathbf{M}_k(\boldsymbol{\theta}^\ell) \succeq 0$ and invoking Proposition 2.38 in [21] yields that $|\theta^\ell_\gamma| \leq \max[M_\ell,\kappa_\ell,1]$ for every $|\gamma| \leq 2k$ and $\ell = 1,\ldots,p$. Consequently, the set of feasible solutions $(\boldsymbol{\theta}^\ell, \boldsymbol{y})$ of (10) is bounded, hence compact. This implies that (10) has an optimal solution $((\boldsymbol{\theta}^{*\ell}), \boldsymbol{y}^*)$. Notice that among the constraints $L_{\boldsymbol{\theta}^{*\ell}}(h_{\alpha\beta}) \geq 0$ are the constraints $L_{\boldsymbol{\theta}^{*\ell}}(g_j) \geq 0$ for all $j \in J_\ell$. As f^ℓ and $-g_j$ are SOS-convex, invoking Lemma 1 yields

$$f^\ell(\mathbf{x}^*_\ell) \;\leq\; L_{\theta^{*\ell}}(f^\ell) \quad\text{and}\quad 0 \;\leq\; L_{\boldsymbol{\theta}^{*\ell}}(g_j) \;\leq\; g_j(\mathbf{x}^*_\ell), \quad \forall j\in J_\ell, \quad \ell = 1,\ldots,p,$$

where $\mathbf{x}^*_\ell := (L_{\boldsymbol{\theta}^{*\ell}}(x_i)) \in \mathbf{K}_\ell$, $i \in I_\ell$, $\ell = 1,\ldots,p$. In addition, the constraint $y^*_\gamma = \theta^{*\ell}_\gamma$, for all ℓ such that $\gamma \in \mathcal{I}^{d_{\max}}_\ell$, implies that $(x^*_\ell)_i = (x^*_{\ell'})_i$ whenever $i \in I_\ell \cap I_{\ell'}$. Therefore defining $x^*_i := (x^*_\ell)_i$ whenever $i \in I_\ell$, one obtains $g_j(\mathbf{x}^*) \geq 0$ for all j, i.e., $\mathbf{x}^* \in \mathbf{K}$. Finally,

$$f^* \geq \tilde{q}^k_1 \;=\; L_{\boldsymbol{y}^*}(f) \;=\; \sum_{\ell=1}^p L_{\boldsymbol{\theta}^{*\ell}}(f^\ell) \;\geq\; \sum_{\ell=1}^p f^\ell(\mathbf{x}^*_\ell) \;=\; f(\mathbf{x}^*),$$

which shows that $\mathbf{x}^* \in \mathbf{K}$ is an optimal solution of (1). Hence $f^* = f(\mathbf{x}^*) = \tilde{q}^k_1$. □

3.2 Sufficient condition for finite convergence

By looking at the dual (10) of the semidefinite program (9) one obtains a sufficient condition for finite convergence. Choose $\omega \in \mathbb{N}$ minimal such that $2\omega \geq \max\{\deg(f), \deg(g_1),\ldots,\deg(g_m)\}$. We have the following lemma.

Lemma 2. *Let $((\boldsymbol{\theta}^{*1},\ldots,\boldsymbol{\theta}^{*p},\boldsymbol{y}^*) \in \mathbb{R}^{s(1,d_{max})}\times\cdots\times\mathbb{R}^{s(p,d_{max})}\times\mathbb{R}^s$ be an optimal solution of (10). If $\operatorname{rank}\mathbf{M}_\omega(\boldsymbol{\theta}^\ell) = 1$ for every $\ell = 1,\ldots,p$, then $\tilde{q}^k_d = f^*$ and $\mathbf{x}^* = (y^*_\gamma)_{|\gamma|=1}$ is an optimal solution of problem (P).*

Proof. If $\operatorname{rank}\mathbf{M}_\omega(\boldsymbol{\theta}^\ell) = 1$, then $(\theta^\ell_\gamma)_{|\gamma|\leq 2\omega}$, is the vector of moments (up to order 2ω) of the Dirac measure $\delta_{\mathbf{x}^\ell}$ at the point $\mathbf{x}^\ell := (\theta^\ell_\gamma)_{|\gamma|=1} \in \mathbb{R}^{n_\ell}$. The constraints $y^*_\gamma = \theta^{*\ell}_\gamma$ for all ℓ such that $\gamma \in \mathcal{I}^{d_{\max}}_\ell$, $\gamma \in \bigcup_{i=1}^p \mathcal{I}^{d_{\max}}_i$, imply that

$$y^{\ell_1}_i \;=\; y^{\ell_2}_i, \quad \forall i \in I_{\ell_1}\cap I_{\ell_2}, \quad \ell_1,\ell_2 = 1,\ldots,p.$$

Hence, $\mathbf{x}^* := (y^*_\gamma)_{|\gamma|=1}$ is well defined. Consequently, for all $q \in \mathbb{R}[\mathbf{x}; I_\ell]_{2\omega}$

$$q(\mathbf{x}^*) = q(\mathbf{x}^\ell) = \int q\;\delta_{\mathbf{x}^\ell} = L_{\boldsymbol{\theta}^{*\ell}}(q) \;=\; L_{\boldsymbol{y}^*}(q) \;=\; \sum_\gamma \mathrm{q}_\gamma\, y^\ell_\gamma.$$

Let $j \in J_\ell$. The constraints $L_{\boldsymbol{\theta}^{*\ell}}(h_{\alpha\beta}) \geq 0$ imply in particular $L_{\boldsymbol{\theta}^{*\ell}}(g_j) \geq 0$. Since $\deg(g_j) \leq 2\omega$, $0 \leq L_{\boldsymbol{\theta}^{*\ell}}(g_j) = g_j(\mathbf{x}^\ell) = g_j(\mathbf{x}^*)$, and so as $j \in J_\ell$ was arbitrary, $\mathbf{x}^* \in \mathbf{K}$. Finally, and again because $\deg(f) \leq 2\omega$,

$$f^* \;\geq\; \tilde{q}^k_d \;=\; L_{\boldsymbol{y}^*}(f) \;=\; f(\mathbf{x}^*),$$

from which we may conclude that $\mathbf{x}^* \in \mathbf{K}$ is an optimal solution of problem (P) in (1). □

4 Computational issues

4.1 Comparing coefficients

As already outlined earlier we implemented polynomial equalities by comparison of coefficients. The resulting constraints in the SDP are sparse and can be treated efficiently by the solver. A crucial point for the implementation of sparse BSOS hence is how to equate the coefficients. The bottle neck for such an implementation is that one has to gather all occurrences of the same monomials.

As in [14], we use the following data format for representing a polynomial f in n variables:

$$F(i, 1:n+1) = [\gamma^T, \mathrm{f}_\gamma],$$

stating that f_γ is the ith coefficient of f corresponding to the monomial $\mathbf{x}^\gamma$. Adding two polynomials is done by concatenating their representations. Hence, equating the coefficients of $\mathbf{x}^\gamma$ is basically finding all indices i of a polynomial F, such that $F(i, 1:n) = \gamma^T$.

Matlab is providing the function `ismember(A,B,'rows')` to find a row `A` in a Matrix `B`. This however is too slow for our purpose. Instead of using this function, we reduce the problem to finding all equal entries of a vector, which can be handled much more efficiently. To that end we multiply $F(:, 1:n)$ by a random vector. Generically this results in a vector whose entries are different if and only if the corresponding rows in $F(:, 1:n)$ are different.

4.2 Reducing problem size

By looking at (9) more closely one may reduce the number of free variables and the number of constraints. It is likely that there are some indices $i \in \{1, \ldots, n\}$, that only appear in one of the I_ℓ, say $i \in I_{\ell_i}$. Hence, for all $\gamma \in \bigcup_{j=1}^{p} \mathcal{I}_j^{d_{\max}}$ such that $\gamma_i \neq 0$ the second equality constraint in (9) reduces to $\mathrm{f}_\gamma^{\ell_i} = f_\gamma$. Consequently, there is a number of variables that are or can be fixed from the beginning. We do this in our implementation. However, in order to be able to certify optimality by Lemma 2, one needs to trace back these substitutions, to recover the moment sequences $\boldsymbol{y}^\ell$ from the solution of the dual problem. Removing these fixed variables occasionally leads to equality constraints $0 = 0$ in the SDP. We remove those constraints for better conditioning.

5 Numerical experiments

In the following section we provide some examples to illustrate the performance of our approach. The examples are chosen to show weaknesses and strengths of the sparse BSOS (SBSOS) hierarchy in comparison to the dense version (BSOS), to demonstrate the effects of different sparsity patterns, and to present the performance on sparse non linear test functions from the literature [25].

Of course to compare the dense and sparse versions we either do it on examples with a relatively small number of variables (in which case k can be larger than 2 for the dense version) or on examples with a larger number of variables (e.g. with $n = 90$ variables) but then with $k = 1$ because for $k > 1$ the size of the semidefinite constraints in the dense version is too large. When we compare both versions we are interested in several issues:

- the influence of the block sizes (depending on the sparsity pattern) when the size of overlaps between blocks of variables is fixed.
- the influence of various block and overlap sizes for a fixed number of variables (e.g. $n = 45$).

- does the finite convergence of the dense version occur systematically earlier than for the sparse version? (As it cannot occur later.)

Of course those observations are biased by the (limited) sample of examples that we have considered. Therefore they should be understood as partial indications rather than definite conclusions. The latter would require much more computational experiments beyond the scope of the present paper.

All experiments were performed on an Intel Core i7-5600U CPU @ 2.60GHz × 4 with 16GB RAM. Scripts are executed in Matlab 8.5 (R2015a) 64bit on Ubuntu 14.04 LTS operating system. The SDP solver used for BSOS and SBSOS is SDPT3-4.0.

The results are presented in tables below. They provide the following informations:

- A pattern or problem code to identify the example.
- The relaxation order d and the chosen parameter k for the SDP constraints.
- The maximal degree $d_{\max}$, appearing in the certificate.
- The result, i. e. the primal solution of the SDP.
- The time in seconds, including the times to generate and solve the SDP as well as computing the optimality condition.
- The abbreviation rk stands for the rank of the moment matrix according to Section 3.2. In the case of SBSOS, rk is the average rank of all moment matrices.
- For the examples marked with * the solver stopped because steps were too short, the maximum number of iterations was achieved, or lack of progress. In these cases one has to consider the result carefully.

5.1 Comparison to the dense version on non sparse examples

As already pointed out in Section 3.1 there is a difference between the BSOS and SBSOS implementation in how equality constraints are handled. While the implementation of BSOS in [14] uses point evaluation, the present implementation of SBSOS uses comparison of coefficients. As a consequence, we expect the sparse version to perform very well on problems of low degree, while we expect it to run into difficulties when the degree of the involved polynomials is high. In particular, if the degree of the constraints is high then $d_{\max}$ grows rapidly with the relaxation order d.

All problems presented in Table 1 are taken from [14]. The first number in the name of each example refers to the number of variables, the second number to the degree of the objective functions and to the maximum degree appearing in the constraints.

Like the dense version, the sparse version is able to find and verify the optimum numerically in all quadratic examples and in some quartic examples. While the sparse verions outperforms the dense on if the degree is low, for higher relaxation orders d it is slower than the dense one. This is due to comparison of coefficients which is fast for low degree and small blocks of variables, but slow if $d_{\max}$ is high. In Table 1, we observe this effect when $d_{\max} \geq 18$.

Recall that the dense version considers a wider class of certificates than the sparse version. Hence, though both hierarchies converges, it might be possible, that BSOS is able to find and verify the optimal value in an earlier relaxation step than SBSOS. In the examples considered here, this does not happen. The sparse version was always (up to numerical errors) able to find the optimum in the same relaxation as the dense version.

			BSOS			SBSOS		
Problem	(d,k)	$d_{\max}$	Result	rk	Time(s)	Result	rk	Time(s)
P4_2	(1,1)	2	-5.7491e-01	1	0.9	-5.7491e-01	1	0.6
P4_4	(1,2)	4	-6.5919e-01	7	0.6	-6.5919e-01	7	0.5
	(2,2)	8	-4.3603e-01	1	1.0	-4.3603e-01	1	0.8
P4_6	(1,3)	6	-6.2500e-02	26	1.0	-6.2500e-02	15	0.7
	(2,3)	12	-6.0937e-02	7	1.0	-6.0938e-02	7	0.9
	(3,3)	18	-6.0693e-02	4	2.9	-6.0693e-02	4	5.7
P4_8	(1,4)	8	-9.4257e-02*	46	13.8	-9.3355e-02	20	2.7
	(2,4)	16	-8.5813e-02	9	3.3	-8.5813e-02	9	1.8
	(3,4)	24	-8.5813e-02	4	5.3	-8.5817e-02*	5	9.7
P6_2	(1,1)	2	-5.7491e-01	1	0.4	-5.7491e-01	1	0.4
P6_4	(1,2)	4	-5.7716e-01	13	1.0	-5.7716e-01	13	0.6
	(2,2)	8	-5.7696e-01	4	4.8	-5.7696e-01	4	4.8
	(3,2)	12	-5.7696e-01	3	23.6	-5.7709e-01*	4	32.7
P6_6	(1,3)	6	-6.5972e-01	36	8.7	-6.5972e-01	35	4.1
	(2,3)	12	-6.5972e-01	32	22.9	-6.5972e-01	32	7.4
	(3,3)	18	-4.1288e-01	1	40.7	-4.1288e-01	1	144.8
P8_2	(1,1)	2	-5.7491e-01	1	0.5	-5.7491e-01	1	0.4
P8_4	(1,2)	4	-6.5946e-01	21	2.2	-6.5946e-01	21	1.2
	(2,2)	8	-4.3603e-01	1	19.5	-4.3603e-01	1	3.4
P10_2	(1,1)	2	-5.7491e-01	1	0.5	-5.7491e-01	1	0.4
P10_4	(1,2)	4	-6.5951e-01	31	8.5	-6.5951e-01	31	3.0
	(2,2)	8	-4.3603e-01	1	28.0	-4.3604e-01*	1	12.4
P20_2	(1,1)	4	-5.7492e-01*	1	1.1	-5.7491e-01	1	0.7

Table 1: Comparison to the dense version on non sparse examples

5.2 Randomly generated sparse functions

The strength of SBSOS is to handle polynomials efficiently when some structured sparsity pattern is present in the data. As a consequence of comparing coefficients for the equality constraints, SBSOS is most powerful for objective functions of degree 2. Hence, given a chosen sparsity pattern, we randomly create quadratic test functions and let the feasible set $\mathbf{K}$ be the product of simplices (with possible overlaps) or intersections of the unit sphere with the positive orthant, respectively[3]. More formally, for a sparsity pattern $I = \{I_1, \ldots, I_p\}$ we consider the problem

$$\begin{aligned} \min \Big\{ x^T A x + b^T x \; : 0 \leq 1 - \sum_{i \in I_\ell} x_i \leq 1, \left(0 \leq 1 - \sum_{i \in I_\ell} x_i^2 \leq 1 \right), \ell = 1, \ldots, p, \\ 0 \leq x_i \leq 1, \quad i = 1, \ldots, n, \} \end{aligned} \tag{11}$$

where b is a random vector and the symmetric matrix A is randomly generate according to I. Depending on the example we either choose the constraint polynomials $1 - \sum_{i \in I_\ell} x_i$, $\ell = 1, \ldots, p$ or the constraint polynomials $1 - \sum_{i \in I_\ell} x_i^2$, $\ell = 1, \ldots, p$.

In this section the sparsity pattern is generated by two vectors $\mathbf{n} \in \mathbb{N}^p$ and $\mathbf{o} \in \mathbb{N}^{p-1}$. The vector $\mathbf{n}$ determinates the size of the blocks I_ℓ whereas the vector $\mathbf{o}$ defines the number of overlapping variables between two consecutive blocks. Defining $c_1 := n_1$ and $c_\ell := c_{\ell-1} + n_\ell - o_{\ell-1})$ we construct

$$I_\ell := \{c_\ell - n_\ell + 1, \ldots, c_\ell\}.$$

Note that the total number of variables in pattern I is c_p. The matrix $A \in \mathbb{R}^{c_p \times c_p}$ is a randomly generated symmetric matrix whose entries are zero outside the submatrices $(A_{ij})_{i,j \in I_\ell}$, $\ell = 1, \ldots, p$. After generating the matrix randomly, we make sure that A has some positive and some negative eigenvalues. Hence, the resulting objective function is not convex.

5.2.1 Comparison to dense version on random sparse examples

Problem I: The problems presented in Table 2 and 3 were chosen to demonstrate the influence of a known sparsity pattern. In particular we are interested in the behaviour of SBSOS for different block sizes in the pattern. To that end for the Problems I we choose a constant number of variables and a constant size of the overlap of to consecutive blocks. Starting with the patterns generated by

$$\mathbf{n} := (5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5) \quad \mathbf{o} := (1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$$

for QPIa ($n = 45$, overlap 1) and

$$\mathbf{n} := (10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10) \quad \mathbf{o} := (2, 2, 2, 2, 2, 2, 2, 2, 2, 2)$$

for QPIb ($n = 90$, overlap 2), we deduce more general patterns of larger block sizes, that are still valid for QPIa and QPIb, respectively. In the tables we only provide the vector $\mathbf{n}$ since the overlap in these problems is fixed. We use an intuitive notation to refer to the vectors $\mathbf{n}$. For example we write (11x5) for $(5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5)$ and (2x17,13) for $(17, 17, 13)$.

[3]Note that Assumption 2 is fulfilled only in the latter case. In the case of simplices one could satisfy the assumption by adding the redundant constraints $M_\ell - \sum_{i \in I_\ell} x_i^2$. However, with same arguments as in the first part of the proof of Theorem 3, one can show that in the specific case of simplex constraints and $d \geq 2$, the feasible set of (10) is compact and an optimal solution is attained. Furthermore, since a simplex is a polyhedron, the quadratic modules associated to the K_ℓ are archimedean. One can adapt the proofs of Theorem 1 and 2 so that the convergence result is still true in this case.

	BSOS	SBSOS	SBSOS	SBSOS	SBSOS	SBSOS	SBSOS
n	45	45	(25,21)	(2x17,13)	(3x13,9)	(5x9,5)	(11x5)
Time(s)	37.82	5.77	2.24	1.65	1.24	1.07	0.73

Table 2: QPIa, $n = 45$, overlap 1, linear constraints, $(d, k) = (2, 1)$, $d_{\max} = 2$, optimality verified in all cases by rank one condition

	BSOS	SBSOS	SBSOS	SBSOS	SBSOS	SBSOS
n	90	90	(50,5x10)	(50,42)	(50,26,18)	(50,2x18,10)
Time(s)	1208.9	73.39	24.50	20.73	19.42	20.28
	SBSOS	SBSOS	SBSOS	SBSOS	SBSOS	SBSOS
n	(2x34,26)	(34,3x18,10)	(3x26,18)	(2x26,2x18,10)	(5x18,10)	(11x10)
Time(s)	9.76	7.52	7.19	6.28	4.59	1.58

Table 3: QPIb, $n = 90$, overlap 2, linear constraints, $(d, k) = (2, 1)$, $d_{\max} = 2$, optimality verified in all cases by rank one condition

Both QPI we constrain to the product of intersecting simplices. Hence, all constraints are linear. We choose parameter $k = 1$. According to the footnote in Section 5.2, we have to choose $d \geq 2$ to guarantee the existence of a dual solution. In our experiments, $d = 2$ already led to the optimal solution. In both examples, QPIa and QPIb, for all sparsity patterns, the optimal value was certified by the rank one condition. Hence, we only display the timings.

Looking at QPIa in Table 2, one observes that SBSOS is able to handle this sparse problem much more efficiently then BSOS, even if no information on the sparsity pattern is given (see column $\mathbf{n} = 45$). Apart from the reasons already given in Section 5.1, here we observe the effect of removing fixed variables from the SDP before solving it (see Section 4.2).

As a smaller block size in the sparsity pattern directly results in a smaller size of the SDP blocks in the relaxation, the solving time decreases when the sparsity pattern of the problem is given more accurately, which is illustrated nicely in the table.

While the sparsity patterns in QPIa were chosen to get blocks of the same size (plus one block for the remaining variables), in QPIb we are interested in the influence of the biggest block of variables. In Table 3 one may observe, that significant improvements in the timing are only achieved, when the size of the biggest block of the pattern is reduced.

Problem II: With the problems in Table 4 we intend to demonstrate the influence of different overlaps. To that end, the vector $\mathbf{n} = (50, 50)$ is fixed and we vary the vector $\mathbf{o}$, which in this example consist of one number determinating the overlap of the two blocks. Consequently, the number of variables in the different examples changes. Since $\mathbf{n}$ is fixed for all examples the problems are determinated by the number of overlapping variables $\mathbf{o}$, which is displayed in the table. In these problems we constrain the feasible set to be the product of intersections of the unit circle with the positive orthant. Again we choose the parameter $k = 1$. In the presented experiments the first relaxation $d = 1$ reaches the optimal value of the polynomial optimization problems, certified by the rank one condition. Hence, we only present the overlap $\mathbf{o}$, the number n of variables, and the timings to compare the examples.

The results show that the increasing number of variables makes the problems more difficult for BSOS, since the size of the SDP variables in BSOS depend on the total number of variables. In contrast to that, the SDP variables in SBSOS have constant size in all experiments. SBSOS even profits from the higher number of variables, as there is less correlation between the two

Overlap		BSOS			SBSOS		
o	n	Result	rk	Time(s)	Result	rk	Time(s)
40	60	-3.7633	1	17.74	-3.7633	1	24.13
30	70	-4.1162	1	31.76	-4.1162	1	17.75
20	80	-4.4683	1	57.85	-4.4683	1	15.11
10	90	-4.7746	1	90.23	-4.7746	1	11.86
5	95	-5.0743	1	122.04	-5.0743	1	11.00
1	99	-5.2489	1	144.71	-5.2489	1	10.40

Table 4: QPII, $\mathbf{n} = (50, 50)$, quadratic constraints, $(d, k) = (1, 1)$, $d_{\max} = 2$, optimality verified in all cases by rank one condition

n. of	block-	over-		SBSOS		n. of	block-	over-		SBSOS	
blocks	size	lap	n	rk	Time(s)	blocks	size	lap	n	rk	Time(s)
500	3	1	1001	1	13.8	1000	3	1	2001	1	40.5
	4	2	1002	1	20.9		4	2	2002	≈ 1	67.1
	5	3	1003	1	31.0		5	3	2003	1	100.0
	6	4	1004	1	44.3		6	4	2004	1	154.5
1500	3	1	3001	1	79.4	500	8	2	3002	1	127.5
	4	2	3002	1	135.5		9	3	3003	1	205.4

Table 5: QPLS, linear constraints, $(d, k) = (2, 1)$, $d_{\max} = 2$,

blocks of variables. This explains, why BSOS is slowing down with decreasing overlap, while SBSOS fastens up.

5.2.2 Performance on a large scale problem

We employ the quadratic problem in (11) to show the range of SBSOS for large scale problems. To this end, we fix an overlap of 1 and consider instances of the quadratic problem for a large number of blocks with a rather small blocksize. This results in problems with total number of variables between 1001 and 3003. We constrain each block of variables to the unit simplex. Choosing parameter $k = 1$, the second relaxation $d = 2$ is sufficient to reach the optimal solution verified by the rank one condition in almost all cases.

In Table 5 we show that SBSOS is able to solve quadratic non convex problems with 3000 variables in reasonable time.[4]

5.3 Test problems from the literature

In this section we present experiments on some test problems considered to be challenging in non linear optimization. All test functions are sums of squares and share the global minimum 0. Hence, it would be possible to compute the minimum in the unconstraint case. However, if not using constraints, the SBSOS approach reduces to searching for sums of squares and does not

[4]Actually we are able to solve problems with even more variables. However, due to problems with the memory management between MATLAB and Linux, we have to proceed in two steps. First we generate the SDP problem, which requires a lot of memory. Once the SDP has been generated, we save it and clear the whole workspace. A "`clear all`" command does only delete all variables inside Matlab. However for some technical reason this newly available memory space cannot be used. So we have to restart Matlab and then load and solve the SDP. In doing this two-step process we can solve QPLS problems with up to 5001 variables in less than 500 seconds.

differ from other sparse hierarchies such as presented in [33]. Hence, we restrict the problems to a convex or non convex compact feasible set $\mathbf{K}$ defined by the $\mathbf{K}_\ell$ according to (6) and for which the global minimizer $\mathbf{x}^*$ belongs to $\mathbf{K}$.

Consider the following test functions, all of degree 4:

- The *Chained Singular Function*:

$$f := \sum_{j\in J} \Big((x_j + 10x_{j+1})^2 + 5(x_{j+2} - x_{j+3})^2 + (x_{j+1} - 2x_{j+2})^4 + 10(x_j - x_{j+3})^4 \Big)$$

  where $J := \{2i-1 \,:\, i = 1, \ldots, n/2-1\}$ and $n \equiv 0 \mod 4$. To fulfil the sparsity assumptions we choose the sparsity pattern $I := \{\{1,2,3,4\}, \{3,4,5,6\}, \ldots\}$. We consider the convex feasible set given by

$$\mathbf{K}_\ell := \left\{ \mathbf{x} \in \mathbb{R}^{n_\ell} \,:\, 0 \le \frac{2}{3} x_i \le 1, i \in I_\ell, 0 \le 1 - \frac{1}{4} \sum_{i\in I_\ell} x_i^2 \le 1 \right\}.$$

- The *Chained Wood Function*:

$$\begin{aligned} f := \sum_{j\in J} \Big(& 100(x_{j+1} - x_j^2)^2 + (1 - x_j)^2 + 90(x_{j+3} - x_{j+2}^2)^2 \\ & + (1 - x_{j+2})^2 + 10(x_{j+1} + x_{j+3} - 2)^2 + 0.1(x_{j+1} - x_{j+3})^2 \Big) \end{aligned}$$

  where $J := \{2i-1 \,:\, i = 1, \ldots, n/2-1\}$ and $n \equiv 0 \mod 4$. The sparsity pattern is the same as for the Chained Singular Function. We consider the non convex feasible set given by

$$\mathbf{K}_\ell := \left\{ \mathbf{x} \in \mathbb{R}^{n_\ell} \,:\, 0 \le x_i \le 1, i \in I_\ell, 0 \le 4 - \sum_{i\in I_\ell} x_i^2 \le 1 \right\}.$$

- The *Generalized Rosenbrock Function*:

$$f := \sum_{i=2}^{n} \Big(100(x_i - x_{i-1}^2)^2 + (1 - x_i)^2 \Big).$$

  The sparsity pattern is $I := \{\{1,2\}, \{2,3\}, \ldots\}$. We consider the non convex feasible set given by

$$\mathbf{K}_\ell := \left\{ \mathbf{x} \in \mathbb{R}^{n_\ell} \,:\, 0 \le x_i \le 1, i \in I_\ell, 0 \le 2 - \sum_{i\in I_\ell} x_i^2 \le 1 \right\}.$$

In Table 6 we show the results of SBSOS for the Chained Singular and the Chained Wood Function of size n from 500 to 1000. For both examples we reach the optimality condition in the second step of the relaxation. Note, that $d_{\max}$ does not increase from the first to the second relaxation step, since the parameter $k = 2$ dominates the total degree of the certificate for small degree in the constraints and low relaxation orders d. Notice also, that the accuracy for the Chained Singular Function is very good, while the results for the Chained Wood Function are fair for a large scale problem.

In Table 7 we provide results for the Generalized Rosenbrock Function for n from 100 to 700 variables. Again we choose the parameter $k = 2$. As for the Chained Wood Function SBSOS reaches the minimum in the first relaxation step. However, for the Rosenbrock Function we need to go to relaxation order $d = 3$ to certify optimality by the rank one condition.

Next we consider the following test functions of degree 6:

			Chained Singular			Chained Wood		
n	(d,k)	$d_{\max}$	Result	rk	Time(s)	Result	rk	Time(s)
500	(1,2)	4	-2.30e-02*	≈4	41.1	-3.05e-05	4	23.25
	(2,2)	4	-3.34e-09	1	33.8	-6.23e-05	1	26.28
600	(1,2)	4	-6.68e-03*	≈4	141.4	-3.73e-05	4	28.87
	(2,2)	4	-3.55e-09	1	43.4	-1.02e-04	1	31.91
700	(1,2)	4	-1.91e-03*	≈2	150.1	-7.83e-05	4	32.46
	(2,2)	4	-1.89e-09	1	51.0	-1.11e-04	1	38.92
800	(1,2)	4	-1.37e-02*	≈4	161.8	-9.35e-05	4	38.47
	(2,2)	4	-2.16e-09	1	62.9	-1.29e-04	1	45.69
900	(1,2)	4	-1.11e-02*	≈4	184.3	-1.07e-04	4	44.67
	(2,2)	4	-2.21e-09	1	71.9	-1.22e-04	1	54.56
1000	(1,2)	4	-2.26e-02*	≈4	206.4	-1.37e-04	4	49.92
	(2,2)	4	-2.81e-09	1	80.0	-1.39e-04	1	62.03

Table 6: Chained Singular and Chained Wood Function (block size: 4)

			Generalized Rosenbrock		
n	(d,k)	$d_{\max}$	Result	rk	Time(s)
100	(1,2)	4	-1.68e-06	≈1	2.66
	(2,2)	4	-3.12e-06	≈1	2.85
	(3,2)	6	-3.30e-06	1	5.65
200	(1,2)	4	-3.96e-06	≈1	4.00
	(2,2)	4	-4.78e-06	≈1	5.09
	(3,2)	6	-6.38e-06	1	11.96
300	(1,2)	4	-5.52e-06	≈1	6.27
	(2,2)	4	-6.71e-06	≈1	7.90
	(3,2)	6	-1.44e-05	1	19.82
400	(1,2)	4	-6.71e-06	≈1	8.89
	(2,2)	4	-1.19e-05	≈1	10.76
	(3,2)	6	-1.46e-05	1	29.78
500	(1,2)	4	-9.34e-06	≈1	10.11
	(2,2)	4	-1.12e-05	≈1	14.12
	(3,2)	6	-2.37e-05	1	39.19
600	(1,2)	4	-1.02e-05	≈1	13.06
	(2,2)	4	-1.18e-05	≈1	17.45
	(3,2)	6	-2.32e-05	1	53.18
700	(1,2)	4	-1.25e-05	≈1	15.63
	(2,2)	4	-2.69e-05	≈1	21.66
	(3,2)	6	-2.71e-05	1	68.36

Table 7: Generalized Rosenbrock Function (block size: 2)

		Broyden Banded			Discrete Boundary		
n	(d,k)	Result	rk	Time	Result	rk	Time
7	(1,2)	-9.68e-09	1	13.2	-1.85e-10	1	1.60
9	(1,2)	7.99e-09	1	110.0	-7.15e-10	1	1.16
11	(1,2)	2.61e-08	1	232.1	-1.36e-09	1	1.56
13	(1,2)	-6.38e-06	1	416.4	-8.54e-10	1	1.44
15	(1,2)	-6.92e-07	1	589.1	-9.48e-10	1	1.42

Table 8: Broyden Banded (block size: 7) and Discrete Boundary Value Function (block size: 3)

- The *Discrete Boundary Value Function*:

$$f := \sum_{i=1}^{n} (2x_i - x_{i-1} - x_{i+1} + \frac{1}{2}h^2(x_i + ih + 1)^3)^2,$$

where $h := \frac{1}{n+1}$,$x_0 := 9 =: x_{n+1}$. To fulfil the sparsity assumptions we choose the sparsity pattern $I := \{\{1,2,3\},\{2,3,4\},\ldots\}$. The feasible set is given by

$$\mathbf{K}_\ell := \left\{ \mathbf{x} \in \mathbb{R}^{n_\ell} \,:\, 0 \leq \frac{1}{2}(1 - x_i) \leq 1, i \in I_\ell, 0 \leq 1 - \sum_{i \in I_\ell} x_i^2 \leq 1 \right\}.$$

- The *Broyden Banded Function*:

$$f := \sum_{i=1}^{n} \left(x_i(2 + 10x_i^2) + 1 - \sum_{j \in J_i} (1 + x_j)x_j \right)^2,$$

where $J_i := \{j : j \neq i, \max(1, i-5) \leq j \leq \min(n, i+1)\}$. To fulfil the sparsity assumptions we choose the sparsity pattern $I := \{\{i-5,\ldots,\min(n,i+1)\}, i = 6,\ldots,n\}$. The feasible set is given by

$$\mathbf{K}_\ell := \left\{ \mathbf{x} \in \mathbb{R}^{n_\ell} \,:\, 0 \leq \frac{1}{2}(1 - x_i) \leq 1, i \in I_\ell, 0 \leq 1 - \frac{1}{2}\sum_{i \in I_\ell} x_i^2 \leq 1 \right\}.$$

The Broyden Banded Function is not considered to be a test function for large scale examples [33, 25]. The Discrete Boundary Value Function serves as a reference. In Table 6 we show the results of SBSOS for both test functions of size n from 9 to 15. For both sextic test functions SBSOS could reach the optimality condition at the first relaxation step with a high accuracy. Note that for the Broyden Banded Function the block size is 7 and the maximal degree is 6. In Table 1 we already saw, that SBSOS slows down when block size and maximum degree are both high (see e.g. P6_6). Indeed, already for $n = 15$ it takes more than 500 seconds to solve the example.

Note the remarkable difference in the timings between both functions. Whereas the solving time increases a lot for the Broyden Banded Function, it remains nearly constant for the Discrete Boundary Value Function. This can be explained by the larger size of the SDP variables in the Broyden Banded Function combined with a larger overlap of the blocks. For the Discrete Boundary Value Function a single variable appears in up to three blocks of variables, causing dependencies for the corresponding SDP variables. For the Broyden Banded Function there are variables connecting up to 7 blocks in the sparsity pattern. This causes a stronger coupling of the SDP variables resulting in a more difficult task for the solver.

6 Conclusion

We have provided a sparse version of the BSOS hierarchy [14] so as to handle large scale polynomial optimization problems that satisfy a structured sparsity pattern. The positivity certificates used in the sparse BSOS hierarchy are coming from a sparse version of Stengle Positivstellensatz, also proved in this paper.

We have tested the algorithm on a sample of non convex problems randomly generated as well as on some typical examples from the literature. The results show that the hierarchy is able to solve small scale dense and large scale sparse polynomial optimization problems in reasonable computational time. In all our experiments where the problem size allowed to compare the dense and sparse versions, finite convergence in the latter took place whenever it took place for the former, and moreover at the same relaxation order. This is remarkable, since in principle convergence of the dense version is at least faster than convergence for the sparse version.

Crucial in our implementation is the comparison of coefficients to state that two polynomials are identical (instead of checking their values on a sample of generic points). This limits the application to problems with polynomials of small degree (say less than 4). In particular if some degree in the problem data is at least 6 and the block size is not small, the resulting SDP can become ill conditioned when the relaxation order increases. Depending on the context in which one wants to use the sparse hierarchy, an alternative may be to implement polynomial identities by sampling.

Acknowledgement

The first author is very thankful to the National University of Singapore and Professor Kim-Chuan Toh for their hospitality and financial support during his stay in Singapore.

References

[1] Ahmadi A.A., Majumdar A. *DSOS and SDSOS Optimization: LP and SOCP-Based Alternatives to SOS Optimization*, Proceedings of the 48th Annual Conference on Information Sciences and Systems, Princeton, NJ, pp. 1–5, March 2014.

[2] Ben-Tal A., Nemirovski A. *Lectures on Modern Convex Optimization*, SIAM, Philadelphia, 2001.

[3] Belousov E.G., Klatte D. *A Frank-Wolfe type theorem for convex polynomial programs*, Comp. Optim. Appl. **22**, pp. 37–48, 2002.

[4] Benabbas S., Georgiou K., Magen A., Tulsiani M. *SDP gaps from pairwise independence*, Theory of Computing **8**, pp. 269–289, 2012.

[5] Benabbas S., Magen A. *Extending SDP integrality gaps to Sherali-Adams with applications to Quadratic Programming and MaxCutGain*, in *Integer Programming and Combinatorial Optimization*, Lecture Notes in Computer Science, Springer 2010, pp. 299–312.

[6] Bertsekas D.P., Nedić A., Ozdaglar E. *Convex Analysis and Optimization*, Athena Scientific, Belmont, Massachusetts, 2003.

[7] Chlamtac E., Tulsiani M. *Convex relaxations and integrality gaps*, in *Handbook of Semidefinite, Conic and Polynomial Optimization*, M. Anjos and J.B. Lasserre Eds., Springer, New York, 2012, pp. 139–170.

[8] C.A. Floudas and P.M. Pardalos, *A Collection of Test Problems for Constrained Global Optimization Algorithms*, Lecture Notes in Comput. Sci. 455, Springer-Verlag, Berlin, 1990.

[9] de Klerk E., Laurent M. *On the Lasserre hierarchy of semidefinite programming relaxations of convex polynomial optimization problems*, SIAM J. Optim. **21**, pp. 824–832, 2011.

[10] Handelman D. *Representing polynomials by positive linear functions on compact convex polyhedra*, Pac. J. Math. **132**, pp. 35–62, 1988.

[11] Helton J.W., Nie J. *Semidefinite representation of convex sets and convex hulls*, in *Handbook on Semidefinite, Conic and Polynomial Optimization*, M. Anjos and J.B. Lasserre Eds., Springer, New York, 2012, pp. 77–112.

[12] Henrion D., Lasserre J.B., Lofberg J. *GloptiPoly 3: moments, optimization and semidefinite programming*, Optim. Methods and Softwares**24**, pp. 761–779, 2009.

[13] Krivine J.L. Anneaux préordonnés, J. Anal. Math. **12**, pp. 307–326, 1964.

[14] Lasserre J.B., K. Toh, S. Yang. *A bounded degree SOS hierarchy for Polynomial Optimization*, EURO J. Computational Optimization, in print.

[15] Lasserre J.B. *Global optimization with polynomials and the problem of moments*, SIAM J. Optim. **11**, pp. 796–817, 2001.

[16] Lasserre J.B. *Semidefinite programming vs. LP relaxations for polynomial programming*, Math. Oper. Res. **27**, pp. 347–360, 2002.

[17] Lasserre J.B. *A Lagrangian relaxation view of linear and semidefinite hierarchies*, SIAM J. Optim. **23**, pp. 1742–1756, 2013

[18] Lasserre J.B. *Convergent SDP-relaxations in polynomial optimization with sparsity*, SIAM J. Optim. **17**, pp. 822–843, 2006

[19] Lasserre J.B. *Polynomial programming: LP-relaxations also converge*, SIAM J. Optim. **15**, pp. 383–393, 2004.

[20] Lasserre J.B. *Moments, Positive Polynomials and Their Applications*, Imperial College Press, London, 2009.

[21] Lasserre J.B. *An Introduction to Polynomial and Semi-Algebraic Optimization*, Cambridge University Press, Cambridge, 2015.

[22] Laurent M. *A comparison of the Sherali-Adams, Lovász-Schrijver and Lasserre relaxations for 0-1 programming*, Math. Oper. Res. **28**, pp. 470–496, 2003.

[23] Marshall M. *Representation of non-negative polynomials, degree bounds and applications to optimization*, Canad. J. Math. **61**, pp. 205–221, 2009.

[24] Nie J. *Optimality conditions and finite convergence of Lasserre's hierarchy*, Math. Program. **146**, pp. 97–121, 2014.

[25] Nie J., Demmel J. *Sparse SOS Relaxations for Minimizing Functions that are Summations of Small Polynomials*, SIAM J. Optim. 19, pp. 1534-1558 (2008).

[26] Putinar M. *Positive polynomials on compact semi-algebraic sets*, Ind. Univ. Math. J. **42**, pp. 969–984, 1993.

[27] Sherali H.D., Adams W.P. *A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems*, SIAM J. Discr. Math. **3**, pp. 411–430, 1990.

[28] Sherali H.D., Adams W.P. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*, Kluwer, Dordrecht, MA, 1999.

[29] Stengle G. *A Nullstellensatz and a Positivstellensatz in semialgebraic geometry*, Math. Ann. **207**, pp. 87–97, 1974.

[30] Toh K.C., Todd M.J., Tutuncu R.H. *SDPT3 — a Matlab software package for semidefinite programming*, Optimization Methods and Software **11**, pp. 545–581, 1999.

[31] Toh K.C., Todd M.J., Tutuncu R.H. *Solving semidefinite-quadratic-linear programs using SDPT3*, Mathematical Programming **95**, pp. 189–217, 2003.

[32] F.-H. Vasilescu. *Spectral measures and moment problems*, in *Spectral Theory and Its Applications*, Theta Foundation, Bucharest, Romania, 2003, pp. 173–215.

[33] Waki S., Kim S., Kojima M., Maramatsu M. *Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity*, SIAM J. Optim. **17**, pp. 218–242, 2006.

[34] Zheng X.J., Sun X.L., Li D., Xu Y.F. *On zero duality gap in nonconvex quadratic programming problems*, J. Glob. Optim. **52**, pp. 229–242, 2012.