EURO J Comput Optim (2017) 5:87–117
DOI 10.1007/s13675-015-0050-y

ORIGINAL PAPER

A bounded degree SOS hierarchy for polynomial optimization

Jean B. Lasserre[1] · Kim-Chuan Toh[2] · Shouguang Yang[2]

Received: 24 January 2015 / Accepted: 7 September 2015 / Published online: 22 September 2015

Abstract We consider a new hierarchy of semidefinite relaxations for the general polynomial optimization problem $(P):\ f^* = \min\{f(x) : x \in K\}$ on a compact basic semi-algebraic set $K \subset \mathbb{R}^n$. This hierarchy combines some advantages of the standard LP-relaxations associated with Krivine's positivity certificate and some advantages of the standard SOS-hierarchy. In particular it has the following attractive features: (a) in contrast to the standard SOS-hierarchy, for each relaxation in the hierarchy, the size of the matrix associated with the semidefinite constraint is the same and fixed in advance by the user; (b) in contrast to the LP-hierarchy, finite convergence occurs at the first step of the hierarchy for an important class of convex problems; and (c) some important techniques related to the use of point evaluations for declaring a polynomial to be zero and to the use of rank-one matrices make an efficient implementation possible. Preliminary results on a sample of non convex problems are encouraging.

Keywords Global optimization · Polynomial optimization · convex relaxations · LP and semidefinite hierarchies

Mathematics Subject Classification 90C26 · 90C22

✉ Jean B. Lasserre
lasserre@laas.fr

Kim-Chuan Toh
mattohkc@nus.edu.sg

Shouguang Yang
matyash@nus.edu.sg

1 LAAS-CNRS and Institute of Mathematics, University of Toulouse, LAAS, 31031 Toulouse cédex 4, Toulouse, France

2 Department of Mathematics, National University of Singapore, 10 Lower Kent Ridge Road, Singapore 119076, Singapore

1 Introduction

We consider the polynomial optimization problem:

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

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

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

for some polynomials $g_j \in \mathbb{R}[x]$, $j = 1, \ldots, m$. In order to approximate (and sometimes solve exactly) (P) one may instead solve a hierarchy of *convex relaxations* of (P) of increasing sizes, namely for instance,

• *Semidefinite* relaxations defined in Lasserre (2001) and based on Putinar's certificate of positivity on K Putinar (1993), where the d-th convex relaxation of the hierarchy is a semidefinite program given by

$$\gamma_d = \max_{t, \sigma_j} \left\{ t : f - t = \sigma_0 + \sum_{j=1}^m \sigma_j \, g_j \right\}. \tag{3}$$

The unknowns σ_j are sums of squares (SOS) polynomials with the degree bound constraint, degree$(\sigma_j g_j) \leq 2d$, $j = 0, \ldots, m$, and the expression in (3) is a certificate of positivity on $\mathbf{K}$ for the polynomial $x \mapsto f(x) - t$.

• *LP*-relaxations based on Krivine-Stengle's certificate of positivity on $\mathbf{K}$ Krivine (1964), Stengle (1974), where the d-th convex relaxation of the hierarchy is a linear program given by

$$\theta_d = \max_{\lambda \geq 0, t} \left\{ t : f - t = \sum_{(\alpha, \beta) \in \mathbb{N}_d^{2m}} \lambda_{\alpha\beta} \prod_{j=1}^m \left(g_j^{\alpha_j} \, (1 - g_j)^{\beta_j} \right) \right\}, \tag{4}$$

where $\mathbb{N}_d^{2m} = \{ (\alpha, \beta) \in \mathbb{N}^{2m} : \sum_j \alpha_j + \beta_j \leq d \}$. The unknown are t and the nonnegative scalars $\lambda = (\lambda_{\alpha\beta})$, and it is assumed that $0 \leq g_j \leq 1$ on $\mathbf{K}$ (possibly after scaling) and the family $\{1, g_j\}$ generates the algebra $\mathbb{R}[x]$ of polynomials. Problem (4) is an LP because stating that the two polynomials in both sides of "=" are equal yields linear constraints on the $\lambda_{\alpha\beta}$'s. For instance, the LP-hierarchy from Sherali–Adams RLT Sherali and Adams (1990) and their variants Sherali and Adams (1999) are of this form.

In both cases, (γ_d) and (θ_d), $d \in \mathbb{N}$, provide two monotone nondecreasing sequences of lower bounds on f^* and if $\mathbf{K}$ is compact, then both converge to f^* as one lets d increases. For more details as well as a comparison of such relaxations, the reader is referred to, e.g., Lasserre (2009, 2002) and Laurent (2003), as well as Chlamtac and Tulsiani (2012) for the impact of LP- and SOS-hierarchies on approximation algorithms in combinatorial optimization.

Of course, in principle, one would much prefer to solve LP-relaxations rather than semidefinite relaxations (i.e. compute θ_d rather than γ_d) because present LP-software packages can solve sparse problems with millions of variables and constraints, which is far from being the case for today's semidefinite solvers. Therefore the hierarchy (3) applies to problems of modest size only. However if some sparsity or symmetry is taken into account then specialized variants as in Waki et al. (2006) can handle problems of much larger size. However, on the other hand, the LP-relaxations (4) suffer from several serious theoretical and practical drawbacks. For instance, it has been shown in Lasserre (2002, 2009) that the LP-relaxations *cannot* be exact for most convex problems, i.e., the sequence of the associated optimal values converges to the global optimum only *asymptotically*, and not in finitely many steps. Moreover, the LPs of the hierarchy are numerically ill-conditioned. This is in contrast with the semidefinite relaxations (3) for which finite convergence takes place for convex problems where $\nabla^2 f(x^*)$ is positive definite at every minimizer $x^* \in \mathbf{K}$ (see de Klerk and Laurent 2011, Corollary 3.3) and occurs at the first relaxation for SOS-convex[1] problems (Lasserre 2009, Theorem 3.3). In fact, as demonstrated in recent works of Marshall (2009) and Nie (2014), finite convergence is generic even for non convex problems.

1.1 Contribution

This paper is in the vein of recent attempts in Lasserre (2013) and Ahmadi and Majumdar (2014) to overcome the important computational burden associated with the standard SOS-hierarchy (3). In particular, in Lasserre (2013) we have suggested another hierarchy of convex relaxations which combines some of the advantages of the SOS- and LP- hierarchies (3) and (4). In the present paper we take advantage of attractive features of the SDPT3 solver Toh et al. (1999, 2003) to provide an effective implementation of this new hierarchy. First preliminary tests on a sample of non convex problems are encouraging and suggest that this new hierarchy might be efficient. This new hierarchy is another type of SOS-hierarchy labelled BSOS (for hierarchy with *bounded* degree SOS) with the following attractive features:

• In contrast to the standard SOS-hierarchy (3), for *each* semidefinite program in the hierarchy, the size $\binom{n+k}{n}$ of the semidefinite matrix variable is now fixed, parametrized by an integer k that one fixes in advance. This integer k determines the degree of a certain SOS polynomial (for instance one may fix $k = 2$), whence the label BSOS (for "bounded"-SOS). Recall that in the standard SOS-hierarchy (3) the size of the semidefinite matrix variable is $\binom{n+d}{n}$ with rank d in the hierarchy.

• In contrast to the LP-hierarchy (4), *finite* convergence occurs at the first step in the hierarchy for a large class of convex problems: typically convex problems defined with convex quadratic polynomials or SOS-convex polynomials of degree at most k. Recall that such finite convergence is impossible for the LP-hierarchy (4).

• Just as in the standard SOS-hierarchy (3), there also exists a sufficient condition for finite convergence of the hierarchy. Namely, it suffices to check whether at an

[1] An SOS-convex polynomial is a convex polynomial whose Hessian factors as $L(x)L(x)^T$ for some rectangular matrix polynomial L. For instance, separable convex polynomials are SOS-convex.

optimal solution of the corresponding SDP some associated moment matrix is rank-one.

• Last but not least, to implement this hierarchy one uses important techniques that dramatically alleviate the computational burden associated with a standard (careless) implementation. Namely, to declare that two polynomials are identical one uses that their values are equal on finitely many randomly chosen points (instead of equating their coefficients). In addition, the SDP solver SDPT3 Toh et al. (1999, 2003) can be used to handle efficiently some types of matrices used in our positivity certificate.

Preliminary computational experiments First, we have compared our results with those obtained with the GloptiPoly software Henrion et al. (2009) (devoted to solving the SOS-hierarchy (3)) on a sample of non convex problems with up to 20 variables. For problems with low degree (in the initial data) and/or low dimension we obtain the global optimum, whereas good lower bounds are always obtained for problems with high degree or higher dimension (e.g. problems with degree 4 and up to 20 variables).

Next, we have also tested the LP-hierarchy (4) on a sample of convex problems and as expected the convergence is very poor and the resulting LPs become ill conditioned. In addition, the LP can be expensive to solve as the LP data are typically dense. In contrast, the new hierarchy (with smallest value $k = 1$ of its parameter) converges at the first step even though some of the problems are defined with polynomials of degree larger than 2.

We have also considered a sample of non convex quadratic problems of the form $\inf\{x^T Ax : x \in \Delta\}$, where $\Delta \subset \mathbb{R}^n$ is the canonical simplex, and A is a randomly generated real symmetric matrix with r negative eigenvalues and $n - r$ positive eigenvalues. For all problems that could be solved with GloptiPoly (up to $n = 20$ variables) we obtain the optimal values. For the other problems with $n = 40, 50, 100$ variables, only the first (dense) relaxation of GloptiPoly can be implemented and yields only a lower bound on the global optimum. For those problems, a better lower bound is obtained in a reasonable amount of time by running the BSOS hierarchy.

Finally, we have considered the minimization of quadratic and quartic polynomials (with up to 40 variables) on the Euclidean unit ball intersected with the positive orthant. Again in those examples only the first SDP-relaxation of GloptiPoly can be implemented, providing only a lower bound. In contrast, BSOS solves all problems at step 2 of the hierarchy in a reasonable amount of time.

Of course this new hierarchy of semidefinite relaxations also has its drawbacks (at least in its present version). Namely, some submatrix (of the matrix used to describe the linear equality constraints of the resulting SDP) is fully dense and many of these linear constraints are nearly dependent, which yields a lack of accuracy in the optimal solution when the order of relaxation d is increased.

2 Main result

2.1 Notation and definitions

Let $\mathbb{R}[x]$ be the ring of polynomials in the variables $x = (x_1, \dots, x_n)$. Denote by $\mathbb{R}[x]_d \subset \mathbb{R}[x]$ the vector space of polynomials of degree at most d, which forms a vector space of dimension $s(d) = \binom{n+d}{d}$, with, e.g., the usual canonical basis (x^α) of monomials. Also, denote by $\Sigma[x] \subset \mathbb{R}[x]$ (resp. $\Sigma[x]_d \subset \mathbb{R}[x]_{2d}$) the space of sums of squares (SOS) polynomials (resp. SOS polynomials of degree at most $2d$). If $f \in \mathbb{R}[x]_d$, we write $f(x) = \sum_{\alpha \in \mathbb{N}^n_d} f_\alpha x^\alpha$ in the canonical basis and denote by $\boldsymbol{f} = (f_\alpha) \in \mathbb{R}^{s(d)}$ its vector of coefficients. Finally, let $\mathcal{S}^n$ denote the space of $n \times n$ real symmetric matrices, with inner product $\langle \mathbf{A}, \mathbf{B} \rangle = \text{trace}\, \mathbf{AB}$. 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 $Q(g_1, \ldots, g_m) \subset \mathbb{R}[x]$ generated by polynomials $g_1, \ldots, g_m$, is defined by

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

We briefly recall two important theorems by Putinar (1993) and Krivine (1964), Stengle (1974) respectively, on the representation of polynomials that are positive on $\mathbf{K}$.

Theorem 1 *Let $g_0 = 1$ and $\mathbf{K}$ in (2) be compact.*

(a) *If the quadratic polynomial $x \mapsto M - \|x\|^2$ belongs to $Q(g_1, \ldots, g_m)$ and if $f \in \mathbb{R}[x]$ is strictly positive on $\mathbf{K}$, then $f \in Q(g_1, \ldots, g_m)$.*

(b) *Assume that $0 \le g_j \le 1$ on $\mathbf{K}$ for every j, and the family $\{1, g_j\}$ generates $\mathbb{R}[x]$. If f is strictly positive on $\mathbf{K}$, then*

$$f = \sum_{\alpha, \beta \in \mathbb{N}^m} c_{\alpha\beta} \prod_j \left(g_j^{\alpha_j} \, (1 - g_j)^{\beta_j} \right),$$

for some (finitely many) nonnegative scalars $(c_{\alpha\beta})$.

2.2 The bounded-SOS-hierarchy (BSOS)

Consider the problem

$$(P): \quad f^* = \min\{ f(x) \mid x \in \mathbf{K} \}$$

where $K \subset \mathbb{R}^n$ is the basic semi-algebraic set defined in (2), assumed to be compact. Moreover we also assume that $g_j(x) \le 1$ for all $x \in \mathbf{K}$ and $j = 1, \ldots, m$, and $\{1, g_j\}$ generates the ring of polynomials $\mathbb{R}[x]$. For every fixed $d \in \mathbb{N}$ and $\lambda = (\lambda_\alpha), \alpha \in \mathbb{N}^{2m}_d$, let $L_d(\cdot, \lambda) \in \mathbb{R}[x]$ be the "Lagrangian" polynomial:

$$x \mapsto L_d(x, \lambda) := f(x) - \sum_{(\alpha,\beta) \in \mathbb{N}^{2m}_d} \lambda_{\alpha\beta} \prod_{j=1}^m g_j(x)^{\alpha_j} \, (1 - g_j(x))^{\beta_j}. \tag{5}$$

With $k \in \mathbb{N}$ fixed, consider the family of optimization problems indexed by $d \in \mathbb{N}$:

$$q_d^k := \sup_{\lambda,t} \{ t \mid L_d(x,\lambda) - t \in \Sigma[x]_k,\ \lambda \geq 0 \}, \qquad d \in \mathbb{N}. \tag{6}$$

Observe that when k is fixed, then for each $d \in \mathbb{N}$

- computing q_d^k in (6) reduces to solving a semidefinite program and therefore (6) defines a *hierarchy* of semidefinite programs because $q_{d+1}^k \geq q_d^k$ for all $d \in \mathbb{N}$.
- The semidefinite constraint is associated with the constraint $L_d(x,\lambda) - t \in \Sigma[x]_k$ and the associated matrix has fixed size $\binom{n+k}{n}$, independent of $d \in \mathbb{N}$, a crucial feature for computational efficiency of the approach.
- If $k = 0$ then (6) is the linear program (4) and so $\theta_d = q_d^0 \leq q_d^k$ for all d, k.

Interpretation For any fixed $d \geq 1$, problem (P) is easily seen to be equivalent to the following problem by adding redundant constraints:

$$(\tilde{P}): \quad f^* = \min\{ f(x) \mid h_{\alpha\beta}(x) \geq 0 \ \ \forall\, (\alpha,\beta) \in \mathbb{N}_d^{2m} \},$$

where $\mathbb{N}_d^{2m} = \{(\alpha,\beta) \in \mathbb{N}^{2m} \mid |\alpha| + |\beta| \leq d\}$ and $h_{\alpha\beta} \in \mathbb{R}[x]$ is the polynomial

$$x \ \mapsto \ h_{\alpha\beta}(x) := \prod_{j=1}^m g_j(x)^{\alpha_j} (1 - g_j(x))^{\beta_j}, \qquad x \in \mathbb{R}^n.$$

The Lagrangian dual of $(\tilde{P})$ is given by

$$(\tilde{P}_d^*): \quad \sup_{\lambda} \{ G_d(\lambda) : \lambda \geq 0 \}$$

where the function $G_d(\cdot)$ is given by

$$\lambda \ \mapsto \ G_d(\lambda) := \inf_{x \in \mathbb{R}^n} \{ L_d(x,\lambda) \}, \qquad \lambda \ \geq \ 0,$$

and $L_d(\cdot,\lambda)$ is the Lagrangian function defined in (5).

Now for a fixed λ, the evaluation of $G_d(\lambda)$ is computational intractable. However, let $k \in \mathbb{N}$ be fixed and observe that

$$\begin{aligned} G_d(\lambda) = \inf_{x \in \mathbb{R}^n} L_d(x,\lambda) &= \sup_t \{ t \mid L_d(x,\lambda) - t \geq 0, \quad \forall\, x \} \\ &\geq \sup_t \{ t \mid L_d(x,\lambda) - t \in \Sigma[x]_k \}, \end{aligned}$$

where recall that $\Sigma[x]_k$ is the space of SOS polynomials of degree at most $2k$. Moreover,

$$f^* \ \geq \ \sup_{\lambda \geq 0} G_d(\lambda) \ \geq \ q_d^k, \qquad \forall\, d \in \mathbb{N}.$$

So the semidefinite program (6) can be seen as a tractable simplification of the intractable problem: $\sup_{\lambda\geq 0} G(\lambda)$. The linear program (4) (which is (6) with $k = 0$) is an even more brutal simplification so that $q^k_d \geq q^0_d = \theta_d$ for all d, k. As a matter of fact we have the more precise and interesting result.

Theorem 2 (Lasserre 2013) *Let* $\mathbf{K} \subset \mathbb{R}^n$ *in* (2) *be compact with nonempty interior and* $g_j(x) \leq 1$ *for* $x \in \mathbf{K}$ *and* $j = 1, \ldots, m$. *Assume further that the family* $\{1, g_j\}$ *generates the algebra* $\mathbb{R}[x]$. *Let* $k \in \mathbb{N}$ *be fixed and for each* $d \in \mathbb{N}$, *let* q^k_d *be as in* (6). *Then*

(a) *The sequence* (q^k_d), $d \in \mathbb{N}$, *is monotone nondecreasing and* $q^k_d \to f^*$ *as* $d \to \infty$.

(b) *Moreover, assume that Slater's condition holds, i.e., there exists* $x_0 \in K$ *such that* $g_j(x_0) > 0$ *for all* $j = 1, \ldots, m$. *If* f *and* $-g_j$, $j = 1, \ldots, m$ *are SOS-convex*[2] *polynomials of degree at most* $2k$ *then* $q^k_1 = f^*$, *i.e., finite convergence takes places at the first relaxation in the hierarchy. In particular when* $f, -g_j$ *are convex quadratic polynomials then* $q^1_1 = f^*$.

Proof The first result is a direct application of Theorem 1(b) since for any integer d, k, $f^* \geq q^k_d \geq q^0_d = \theta_d$, and by Theorem 1, $\theta_d \to f^*$ as $d \to \infty$. Next, if Slater's condition holds and f and $-g_j$ are SOS-convex, $j = 1, \ldots, m$, then there exist nonnegative Lagrange-KKT multipliers $\lambda \in \mathbb{R}^m_+$ such that

$$\nabla f(x^*) - \sum_j \lambda_j \nabla g_j(x^*) = 0; \quad \lambda_j\, g_j(x^*) = 0, \quad j = 1, \ldots, m.$$

In other words, the Lagrangian $x \mapsto L(x) := f(x) - f^* - \sum_j \lambda_j\, g_j(x)$ is SOS-convex and satisfies $L(x^*) = 0$ and $\nabla L(x^*) = 0$. By Helton and Nie (2010, Lemma 8, p. 33), L is SOS (of degree at most $2k$), and thus $q^k_1 = f^*$. ◻

2.3 The SDP formulation of (6)

To formulate (6) as a semidefinite program one has at least two possibilities depending on how we state that two polynomials $p, q \in \mathbb{R}[x]_d$ are identical. Either by *equating their coefficients* (e.g. in the monomial basis), i.e., $p_\alpha = q_\alpha$ for all $\alpha \in \mathbb{N}^n_d$, or by *equating their values* on $\binom{n+d}{n}$ generic points (e.g. randomly generated on the box $[-1, 1]^n$). In the present context of (6) we prefer the latter option since expanding the polynomial $h_{\alpha\beta}(x)$ symbolically to get the coefficients with respect to the monomial basis can be expensive and memory intensive.

[2] A polynomial $f \in \mathbb{R}[x]$ is SOS-convex if its Hessian $\nabla^2 f$ is an SOS matrix, i.e., $\nabla^2 f(x) = L(x)\, L(x)^T$ for some matrix polynomial $L \in \mathbb{R}[x]^{n\times p}$ and some $p \in \mathbb{N}$.

Let $\tau = \max\{\deg(f), 2k, d\max_j\{\deg(g_j)\}\}$. Then for k fixed and for each d, we get

$$\begin{aligned} q_d^k &= \sup\Big\{t \mid f(x) - t - \sum_{(\alpha,\beta)\in\mathbb{N}_d^{2m}} \lambda_{\alpha\beta}\, h_{\alpha\beta}(x) = \langle Q, v_k(x)v_k(x)^T\rangle; \\ &\qquad Q \in \mathcal{S}_+^{s(k)}, \lambda \geq 0\Big\} \qquad (7) \\ &= \sup\left\{t \,\middle|\, \begin{array}{l} f(x^{(p)}) = t + \sum_{(\alpha,\beta)\in\mathbb{N}_d^{2m}} \lambda_{\alpha\beta}\, h_{\alpha\beta}(x^{(p)}) + \langle Q, v_k(x^{(p)})v_k(x^{(p)})^T\rangle, \\ p = 1, \ldots, L,\ Q \in \mathcal{S}_+^{s(k)},\ \lambda \geq 0,\ t \in \mathbb{R} \end{array}\right\} \end{aligned} \qquad (8)$$

where $L := |\mathbb{N}_\tau^n| = \binom{n+\tau}{n}$ and $\{x^{(p)} \in \mathbb{R}^n \mid p = 1, \ldots, L\}$ form a set of *generic* points in $[-1, 1]^n$, and $v_k(x)$ is the vector of the monomial basis of $\mathbb{R}[x]_k$, the vector space of polynomials of degree at most k $\left(\text{whose dimension is } s(k) = \binom{n+k}{k}\right)$. Therefore, if the optimal value q_d^k of (7) is finite then every optimal solution (q_d^k, λ^*, Q^*) of (7) is also an optimal solution of (8).

Remark 1 To get a set of generic points one may first start with a set Θ of L points randomly generated, e.g., according to the uniform distribution on $[0, 1]^n$. Then it is well known that with probability one, the resulting collection of equality constraints in (8) is equivalent to the polynomial identity in (7). In principle, once such a set Θ has been generated, the equivalence is guaranteed if the matrix

$$R(\Theta) := \left[\, v_\tau(x^{(1)}), \ldots, v_\tau(x^{(L)}) \,\right]$$

is nonsingular. In the unlikely event that $R(\Theta)$ is singular, one can generate another sample Θ of random points and check again whether the new corresponding matrix $R(\Theta)$ is singular or not. If not, the procedure is repeated until a non-singular matrix $R(\Theta)$ is obtained. However, in practice there is little need to do so since the above matrix $R(\Theta)$ is nonsingular with probability one.

2.4 Sufficient conditions for finite convergence

By looking at the dual of the semidefinite program (8) one obtains sufficient conditions for finite convergence. To describe the dual of the semidefinite program (8) we need to introduce some notation.

For every $p = 1, \ldots, L$, denote by $\delta_{x^{(p)}}$ the Dirac measure at the point $x^{(p)} \in \mathbb{R}$ and let $\langle q, \delta_{x^{(p)}}\rangle = q(x^{(p)})$ for all $p = 1, \ldots, L$, and all $q \in \mathbb{R}[x]$.

With a real sequence $\mathbf{y} = (y_\alpha)$, $\alpha \in \mathbb{N}_{2\ell}^n$, denote by $\mathbf{M}_\ell(\mathbf{y})$ the moment matrix associated with $\mathbf{y}$. It is a real symmetric matrix with rows and columns indexed by $\mathbb{N}_\ell^n$, and with entries

$$\mathbf{M}_\ell(\mathbf{y})(\alpha, \beta) = y_{\alpha+\beta}, \qquad \forall\, \alpha, \beta \in \mathbb{N}_\ell^n.$$

Similarly, for $j = 1, \ldots, m$, letting $g_j(x) = \sum_\gamma g_{j\gamma} x^\gamma$, denote by $\mathbf{M}_\ell(g_j\, \mathbf{y})$ the localizing matrix associated with $\mathbf{y}$ and $g_i \in \mathbb{R}[x]$. Its rows and columns are also indexed by $\mathbb{N}^n_\ell$, and with entries

$$\mathbf{M}_\ell(g_j\, \mathbf{y})(\alpha, \beta) = \sum_{\gamma \in \mathbb{N}^n} g_{j\gamma}\, y_{\alpha+\beta+\gamma}, \qquad \forall\, \alpha, \beta \in \mathbb{N}^n_\ell.$$

Moment and localizing matrices play an important role in semidefinite relaxations of polynomial optimization problems. For more details the interested reader is referred to Lasserre (2009).

The dual of the semidefinite program (8) reads

$$\begin{aligned} \tilde{q}^k_d := \inf_{\theta \in \mathbb{R}^L} &\sum_{p=1}^L \theta_p\, \langle f, \delta_{x^{(p)}} \rangle \\ \text{s.t.} &\sum_{p=1}^L \theta_p\, (v_k(x^{(p)})\, v_k(x^{(p)})^T) \succeq 0 \\ &\sum_{p=1}^L \theta_p\, \langle h_{\alpha\beta}, \delta_{x^{(p)}} \rangle \geq 0, \quad (\alpha, \beta) \in \mathbb{N}^{2m}_d \\ &\sum_{p=1}^L \theta_p = 1. \end{aligned} \tag{9}$$

(Notice that the weights θ_p are not required to be nonnegative.) By standard weak duality in convex optimization, and for every fixed $k \in \mathbb{N}$, one has

$$f^* \geq \tilde{q}^k_d \geq q^k_d, \qquad \forall d \in \mathbb{N}.$$

Next, we have the following *verification* lemma.

Lemma 1 *Let K in (2) be compact with nonempty interior and assume that there exists $x_0 \in K$ such that $0 < g_j(x_0) < 1$ for all $j = 1, \ldots, m$.*

(a) *For every fixed d sufficiently large (say $d \geq d_0$), the semidefinite program (7) (and thus (8) as well) has an optimal solution.*

(b) *Let $s \in \mathbb{N}$ be the smallest integer such that $2s \geq \max[\deg(f); \deg(g_j)]$, and let $r := \max_j \lceil \deg(g_j)/2 \rceil$. Let $\theta^* \in \mathbb{R}^L$ be an optimal solution of (9) (whenever it exists) and let $\mathbf{y}^* = (y^*_\alpha)$, $\alpha \in \mathbb{N}^n_{2s}$, with*

$$y^*_\alpha := \sum_{p=1}^L \theta^*_p\, (x^{(p)})^\alpha, \qquad \alpha \in \mathbb{N}^n_{2s}. \tag{10}$$

- *If $\operatorname{rank} \mathbf{M}_s(\mathbf{y}^*) = 1$ then $\tilde{q}^k_d = f^*$ and $x^* = (y^*_\alpha)$, $|\alpha| = 1$, i.e., $x^* = \sum_{p=1}^L \theta^*_p\, x^{(p)}$, is an optimal solution of problem (P).*

- *If* $\mathbf{M}_s(\mathbf{y}^*) \succeq 0$, $\mathbf{M}_{s-r}(g_j\,\mathbf{y}^*) \succeq 0$, $j = 1, \ldots, m$, *and* $\operatorname{rank} \mathbf{M}_s(\mathbf{y}^*) = \operatorname{rank} \mathbf{M}_{s-r}(\mathbf{y}^*)$, *then* $\tilde{q}_d^k = f^*$, *and problem* (P) *has* $\operatorname{rank} \mathbf{M}_s(\mathbf{y}^*)$ *global minimizers that can be extracted by a linear algebra procedure.*

The proof is postponed to the Appendix. Notice that we do not claim that problem (9) has always an optimal solution. Lemma 1 is a *verification lemma* (or a stopping criterion) based on some sufficient rank condition on $\mathbf{M}_s(\mathbf{y}^*)$ and $\mathbf{M}_{s-r}(\mathbf{y}^*)$, provided that an optimal solution $\mathbf{y}^*$ exists. When the latter condition holds true then $f^* = \tilde{q}_d^k$ and we can stop as at least one global optimal solution $x^* \in K$ has been identified.

2.5 On the rank-one matrices of (6) and SDPT3

Note that in the SDP (8), the constraint matrices associated with Q are all dense rank-1 matrices of the form $A_p = v_k(x^{(p)})v_k(x^{(p)})^T$. Thus if we let $\boldsymbol{v}_p = v_k(x^{(p)})$, then the linear maps involved in the equality constraints of the SDP can be evaluated cheaply based on the following formulas:

$$\mathcal{A}(X) := \Big[\langle A_p, X\rangle\Big]_{p=1}^{L} = \Big[\langle \boldsymbol{v}_p, X\boldsymbol{v}_p\rangle\Big]_{p=1}^{L}, \quad \mathcal{A}^* y := \sum_{y=1}^{L} y_p A_p = V\,\mathrm{Diag}(y)V^T,$$

where $X \in \mathcal{S}^{s(k)}$, $y \in \mathbb{R}^L$, $V = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_L] \in \mathbb{R}^{s(k)\times L}$. Moreover, one need not store the dense constraint matrices $\{A_p \mid p = 1, \ldots, L\}$ but only the vectors $\{\boldsymbol{v}_p \mid p = 1, \ldots, L\}$. *To solve the SDP* (8) *efficiently, we need to exploit the rank-1 structure of the constraint matrices during the iterations.* Fortunately, the SDPT3 solver Toh et al. (1999, 2003) based on interior point methods has already been designed to exploit such a rank-1 structure to minimize the memory needed to store the constraint matrices, as well as to minimize the computational cost required to compute the Schur complement matrix arising in each interior-point iteration. More precisely, in each iteration where a positive definite matrix $W \in \mathcal{S}^{s(k)}$ is given, one needs to compute the Schur complement matrix $\mathbf{S}$ whose (p, q) element is given by

$$\mathbf{S}_{pq} = \langle A_p, WA_qW\rangle = \langle \boldsymbol{v}_p\boldsymbol{v}_p^T, W\boldsymbol{v}_q\boldsymbol{v}_q^T W\rangle = \langle \boldsymbol{v}_p, W\boldsymbol{v}_q\rangle^2, \quad p, q = 1, \ldots, L.$$

It is the combination of these two implementation techniques (point evaluation in the formulation and exploiting rank-one structure in the interior point algorithm) that makes our implementation of the SOS-hierarchy (6) efficient.

3 Computational issues

Given $f \in \mathbb{R}[x]_d$, to efficiently evaluate the vector $f(x^{(p)})$, $p = 1, \ldots, L$, we need a convenient representation of the polynomial $f(x)$. In our implementation of BSOS, we use the following data format to input a polynomial:

$$\boldsymbol{F}(i, 1 : n+1) = [\alpha^T, f_\alpha]$$

where f_α is the ith coefficient corresponding to the monomial x^α. Note that the enumeration of the coefficients of $f(x)$ is not important. For a given point $z \in \mathbb{R}^n$ such that $z_i \neq 0$ for all $i = 1, \dots, n$, we evaluate $f(z)$ via the following procedure written in MATLAB syntax:

Step 1: Set $\boldsymbol{P} = \boldsymbol{F}(:, 1 : n)$, $\boldsymbol{f} = \boldsymbol{F}(:, n + 1)$, and $s = (s_1, \dots, s_n)^T$, where $s_i = 1$ if $z_i < 0$, and $s_i = 0$ if $z_i \geq 0$.
Step 2: Compute $\bar{s} = \texttt{rem}(\boldsymbol{P}s, 2)$ and $\boldsymbol{z} = \exp(\boldsymbol{P} \log |z|)$.
Step 3: Compute $f(z) = \langle \boldsymbol{f}^{(a)}, \boldsymbol{z}^{(a)} \rangle - \langle \boldsymbol{f}^{(b)}, \boldsymbol{z}^{(b)} \rangle$, where $\boldsymbol{f}^{(a)} = \boldsymbol{f}(\texttt{find}(\bar{s} == 0))$ and $\boldsymbol{f}^{(b)} = \boldsymbol{f}(\texttt{find}(\bar{s} == 1))$.

(The above procedure can be modified slightly to handle the case when z has some zero components.) Note that in the above procedure, $\langle \boldsymbol{f}^{(a)}, \boldsymbol{z}^{(a)} \rangle$ and $\langle \boldsymbol{f}^{(b)}, \boldsymbol{z}^{(b)} \rangle$ correspond to the sum of positive terms and sum of negative terms in the evaluation of $f(z)$. By separating the summation of the positive and negative terms in the evaluation of $f(z)$, it is hoped that cancellation errors can be minimized.

We should mention that some of the equality constraints in (8) may be redundant. For the sake of reducing the computational cost and improve the numerical stability, we remove these redundant constraints before solving the SDP. However, as d increases, the linear constraints would become more and more nearly dependent, and typically the SDP problem cannot be solved accurately by either SDPT3 or SEDUMI.

Another numerical issue which we should point out is that the constraint matrix

$$\begin{bmatrix} \left(h_{\alpha\beta}(x^{(1)})\right)_{(\alpha,\beta)\in\mathbb{N}_d^{2m}} \\ \vdots \\ \left(h_{\alpha\beta}(x^{(L)})\right)_{(\alpha,\beta)\in\mathbb{N}_d^{2m}} \end{bmatrix}$$

associated with the nonnegative vector $(\lambda_{\alpha\beta})$ is typically fully dense. Such a matrix would consume too much memory and also computational cost when d increases or when m is large.

4 Numerical experiments

We call our approach BSOS (for hierarchy with bounded degree SOS). As mentioned in the Introduction, we conduct experiments on three classes of problems which will be described in the ensuing subsections.

4.1 Comparison of BSOS with gloptiploy

We construct a set of test functions with 5 constraints. The test functions are mainly generated based on the following two problems:

$$\begin{array}{llllllllll}
(P_1) & f = & x_1^2 & -x_2^2 & & +x_3^2 & -x_4^2 & +x_1 & -x_2 \\
\text{s.t.}\ 0 & \le g_1 = & 2x_1^2 & +3x_2^2 & +2x_1x_2 & +2x_3^2 & +3x_4^2 & +2x_3x_4 & \le 1 \\
0 & \le g_2 = & 3x_1^2 & +2x_2^2 & -4x_1x_2 & +3x_3^2 & +2x_4^2 & -4x_3x_4 & \le 1 \\
0 & \le g_3 = & x_1^2 & +6x_2^2 & -4x_1x_2 & +x_3^2 & +6x_4^2 & -4x_3x_4 & \le 1 \\
0 & \le g_4 = & x_1^2 & +4x_2^2 & -3x_1x_2 & +x_3^2 & +4x_4^2 & -3x_3x_4 & \le 1 \\
0 & \le g_5 = & 2x_1^2 & +5x_2^2 & +3x_1x_2 & +2x_3^2 & +5x_4^2 & +3x_3x_4 & \le 1 \\
0 & \le x. & & & & & & &
\end{array}$$

The optimal value of (P_1) is $f(x^*) = -0.57491$, as computed by GloptiPoly3. For BSOS, we get the result $q_{d=1}^{k=1} = -0.57491$, which is the exact result.

The second problem is

$$\begin{array}{llllll}
(P_2) & f = & x_1^4x_2^2 & +x_1^2x_2^4 & -x_1^2x_2^2 & \\
\text{s.t.}\ 0 & \le g_1 = & x_1^2 & +x_2^2 & & \le 1 \\
0 & \le g_2 = & 3x_1^2 & +2x_2^2 & -4x_1x_2 & \le 1 \\
0 & \le g_3 = & x_1^2 & +6x_2^4 & -8x_1x_2 + 2.5 & \le 1 \\
0 & \le g_4 = & x_1^4 & +3x_2^4 & & \le 1 \\
0 & \le g_5 = & x_1^2 & +x_2^3 & & \le 1 \\
\multicolumn{6}{l}{0 \le x_1, \quad 0 \le x_2.}
\end{array}$$

The optimal value of (P_2) is $f(x^*) = -0.037037$, as computed by GloptiPoly3. The results obtained by BSOS are

$$\begin{array}{lll}
q_{d=1}^{k=3} = -0.041855, & q_{d=2}^{k=3} = -0.037139, & q_{d=3}^{k=3} = -0.037087 \\
q_{d=4}^{k=3} = -0.037073, & q_{d=5}^{k=3} = -0.037046 & \\
q_{d=1}^{k=4} = -0.038596, & q_{d=2}^{k=4} = -0.037046, & q_{d=3}^{k=4} = -0.037040 \\
q_{d=4}^{k=4} = -0.037038, & q_{d=5}^{k=4} = -0.037037. &
\end{array}$$

Based on the above two problems, we increase the degree of the objective function and constraint functions to generate other test instances which are given explicitly in the Appendix.

Table 1 compares the results obtained by BSOS and GloptiPoly3 for the tested instances. We observe that BSOS can give the exact result for those problems with either low degree or low dimension, while also providing a good lower bound for high-degree and high-dimensional problems. In particular on this sample of problems, k is chosen so that the size of the semidefinite constraint $\left(\text{which is } \binom{n+k}{n}\right)$ is the same as the one needed in GloptiPoly, to certify global optimality. Then notice that BSOS succeeds in finding the optimal value even though the positivity certificate used in (8) is not Putinar's certificate (3) used in GloptiPoly. In addition, for most of test problems, BSOS can usually get better bounds as d increases, and in most cases, the bound is good enough for small $d = 2, 3$.

In Table 1, we also use the sufficient condition stated in Lemma 1 to check whether the generated lower bound is indeed optimal. For quite a number of instances, the moment matrix $\mathbf{M}_\ell(\mathbf{y}^*)$ associated with the optimal solution θ^* of (9) indeed has numerical rank equal to one (we declare that the matrix has numerical rank equal to one if the largest eigenvalue is at least 10^4 times larger than the second largest eigenvalue), which certifies that the lower bound is actually the optimal value. We should note that for some of the instances, although the lower bound is actually the optimal value (as declared by GloptiPoly), but the rank of the moment matrix $\mathbf{M}_\ell(\mathbf{y}^*)$ is larger than one.

4.2 Comparison of BSOS with the LP relaxations of Krivine-Stengle on convex problems

Here we compare the performance of BSOS with the LP relaxations of Krivine-Stengle on convex problems where each test problem has 5 constraint functions in addition to the nonnegative constraint $x \geq 0$. Note that the LP relaxation problem has exactly the same form as in (8), except that the positive semidefinite matrix variable Q is set to 0. We should mention that even though the Krivine-Stengle scheme generates LP problems instead of SDP problems, the size of the corresponding LP problems also increases rapidly with d, like for the BSOS scheme. In particular, in both LP- and BSOS-relaxations, the dimension of the nonnegative variable λ is $\binom{2m+d}{d}$, and the constraint matrix is fully dense. (The BSOS-relaxations include an additional semidefinite constraint with fixed matrix size $\binom{n+k}{k}$.) The following example illustrates the performance of LP relaxation method:

$$
\begin{array}{lrcl}
(C_1) & \min f & = & x_1^4 + x_2^4 + 2x_1^2x_2^2 - x_1 - x_2 \\
& \text{s.t. } 0 \leq g_1 & = & -x_1^4 - 2x_2^4 + 1 \\
& 0 \leq g_2 & = & -2x_1^4 - x_2^4 + 1 \\
& 0 \leq g_3 & = & -x_1^4 - 4x_2^2 + 1.25 \\
& 0 \leq g_4 & = & -4x_1^4 - x_2^4 + 1.25 \\
& 0 \leq g_5 & = & -2x_1^4 - 3x_2^2 + 1.1 \\
& 0 \leq x_1 & & \\
& 0 \leq x_2. & &
\end{array}
$$

For this problem, the functions f and $-g_i$'s are all convex. The optimal value for this problem is $f(x^*) = -0.7500$, as computed by GloptiPoly3. For BSOS, we get $q_{d=1}^{k=2} = -0.7500$, and we obtained the exact result by just choosing $d = 1$. This observation is consistent with Theorem 4.1 in Lasserre (2013). For the LP relaxation method, we get the following values for various choices of d:

$$
\begin{array}{l}
q_{d=1}^{LP} = \text{infeasible}, \quad q_{d=2}^{LP} = -1.2200, \\
q_{d=3}^{LP} = -1.0944, \quad q_{d=4}^{LP} = -0.9696, \quad q_{d=5}^{LP} = \text{fail}.
\end{array}
$$

Observe that when d increases, we could get a better lower bound for the exact optimal value. However, as d increases, the LP relaxation problem would become increasing ill-posed and the solver has difficulty in solving LP problem accurately. In particular,

Table 1 Comparison of BSOS and GloptiPoly3

Problem	BSOS				GloptiPoly3		
	(d, k)	Result	Time (s)	Rank$(\mathbf{M}(\mathbf{y}^*))$	Result	Time (s)	Order, optimal
`P4_2`	1, 1	−6.7747e−001	0.3	1	−6.7747e−001	0.2	1, yes
	2, 1	−6.7747e−001	0.5	1			
`P4_4`	1, 2	−2.9812e−001	0.5	7	−3.3539e−002	0.3	2, yes
	2, 2	−3.3539e−002	0.6	4			
`P4_6`	1, 3	−6.2500e−002	0.8	31	−6.0693e−002	0.5	3, yes
	2, 3	−6.0937e−002	0.9	7			
	3, 3	−6.0693e−002	1.8	4			
`P4_8`	1, 4	−9.3354e−002∗	3.2	>10	−8.5813e−002	2.6	4, yes
	2, 4	−8.5813e−002	3.7	9			
	3, 4	−8.5813e−002	5.1	4			
`P6_2`	1, 1	−5.7491e−001	0.3	1	−5.7491e−001	0.2	1, yes
	2, 1	−5.7491e−001	0.8	1			
`P6_4`	1, 2	−5.7716e−001	1.1	10	−5.7696e−001	0.3	2, yes
	2, 2	−5.7696e−001	1.1	4			
	3, 2	−5.7696e−001	4.3	1			
`P6_6`	1, 3	−6.5972e−001	7.1	>10	−4.1288e−001	6.4	3, yes
	2, 3	−6.5972e−001	10.2	>10			
	3, 3	−4.1288e−001	32.0	1			
`P6_8`	1, 4	−6.5973e−001	74.2	>10	−4.0902e−001	207.2	4, yes
	2, 4	−6.5973e−001	168.6	>10			
	3, 4	−6.5973e−001	264.1	>10			
	4, 4	−4.0928e−001∗	1656.0	1*			
8 var, deg 2	1, 1	−5.7491e−001	0.5	1	−5.7491e−001	0.3	1, yes
	2, 1	−5.7491e−001	0.9	1			
8 var, deg 4	1, 2	−6.5946e−001	2.8	>10	−4.3603e−001	1.5	2, yes
	2, 2	−4.3603e−001	4.8	1			
8 var, deg 6	1, 3	−6.5973e−001	127.1	>10	−4.1288e−001	161.3	3, yes
	2, 3	−6.5973e−001	126.6	>10			
	3, 3	−4.1322e−001*	258.7	1*			
10 var, deg 2	1, 1	−5.7491e−001	0.4	1	−5.7491e−001	0.2	1, yes
	2, 1	−5.7491e−001	1.0	1			
10 var, deg 4	1, 2	−6.5951e−001	7.8	1	−4.3603e−001	5.3	2, yes
	2, 2	−4.3603e−001	20.0	1			
	3, 2	−4.3603e−001∗	66.7	1*			
20 var, deg 2	1, 1	−5.7491e−001	1.2	1	−5.7491e−001	0.4	1, yes
	2, 1	−5.7491e−001	3.0	1			
20 var, deg 4	1, 2	Infeasible	302.1	-	−4.3603e−001	5600.8	2, yes
	2, 2	−4.3602e−001*	1942.2	1*			

An entry marked with "∗" means that the corresponding SDP was not solved to high accuracy for $d = 5$, both the solvers SeDuMi and SDPT3 fail to compute an accurate enough solution for the LP to generate a sensible lower bound for $f(x^*)$.

In Table 2, we observe that BSOS can achieve the exact result with $d = 1$ for all the test instances. In contrast, the LP relaxation method of Krivine-Stengle does not perform very well even though the test instances are convex problems. In particular, observe that for the last instance `C20_2`, the LP relaxation method cannot produce a good lower bound even when we choose $d = 3$, and the time taken to solve the correspond LP is about 40 minutes.

4.3 Performance of BSOS on quadratic problems with polyhedral constraints

Here consider the following problem:

$$\begin{array}{l} \min\ x^T A x \\ \text{ s.t. } e^T x \le 1, \quad x \ge 0, \quad x \in \mathbb{R}^n, \end{array} \tag{11}$$

where A is a given $n \times n$ symmetric matrix. In our numerical experiments, we generate random instances such as `Qn10_r2` for which $n = 10$ and A is randomly generated so that it has $r = 2$ negative eigenvalues and $n - r$ positive eigenvalues as follows:

```
rng('default')
A1 = randn(n); A2 = A1*A1'; perm=randperm(n);
[V,D] = eig(A); eigval=diag(D); idx1=perm(1:r); idx2=perm
    (r+1:n);
V1=V(:,idx1); V2=V(:,idx2); d1=eigval(idx1); d2=eigval(idx2);
A =V2*diag(d2)*V2' - V1*diag(d1)*V1';
```

Table 3 compares the performance of BSOS and GloptiPoly3. From the numerical results, we can see that BSOS is far more efficient than GloptiPoly3 in solving the problems (11). For example, for the problem `Qn20_r2` with $n = 20$, BSOS took only 1.9 seconds to generate the lower bound $-2.0356e3$ for the problem, but GloptiPoly3 took more than 1 hour to generate the same bound. The disparity in the efficiency between BSOS and GloptiPoly3 is expected to become even wider for other instances with n larger than 20.

In Table 3, we again use the sufficient condition stated in Lemma 1 to check whether the generated lower bound is indeed optimal. For each of the first eight instances, the moment matrix $\mathbf{M}_\ell(\mathbf{y}^*)$ associated with the optimal solution θ^* of (9) has numerical rank equal to one (we declare that the matrix has numerical rank equal to one if the largest eigenvalue is at least 10^4 times larger than the second largest eigenvalue), which certifies that the lower bound is actually the optimal value.

4.4 Performance of BSOS on higher order problems with more variables

Here we consider the following problem:

Table 2 Comparison of BSOS with LP relaxations of Krivine-Stengle on convex problems

	LP			BSOS			GloptiPoly3		
	d	Result	Time (s)	d, k	Result	Time (s)	Result	Time (s)	Order, optimal
`C4_2`	1	Infeasible		1, 1	−2.5000e−001	0.4	−2.5000e−001	0.2	1, yes
	2	−9.0000e−001	0.1						
	3	−5.8852e−001	0.3						
	4	−4.2500e−001	5.6						
	5	−3.4975e−001	98.4						
	6	−3.1001e−001	4074.1						
`C4_4`	≤ 3	Infeasible		1, 2	−6.9574e−001	0.6	−6.9574e−001	0.2	2, yes
	4	−1.1094e+000	16.9						
	5	−8.8542e−001	788.4						
`C4_6`	≤ 5	Infeasible		1, 3	−1.1933e+000	1.5	−1.1933e+000	0.5	3, unknown
	6	Fail							
`C6_2`	1	Infeasible		1, 1	−2.5000e−001	0.3	−2.5000e−001	0.2	1, yes
	2	−9.0000e−001	0.1						
	3	−5.8852e−001	0.6						
	4	−4.2500e−001	66.3						
	5	−3.4975e−001	4069.3						
`C6_4`	≤ 3	Infeasible		1, 2	−6.9574e−001	1.3	−6.9574e−001	0.4	2, yes
	4	−1.1094e+000	177.5						
`C6_6`	≤ 5	Infeasible		1, 3	−1.1933e+000	1.3	−1.1933e+000	0.4	3, unknown
	6	Out of memory							
`C8_2`	1	Infeasible		1, 1	−2.5000e−001	0.4	−2.5000e−001	0.2	1, yes
	2	−9.0000e−001	0.1						

Table 2 continued

	LP			BSOS			GloptiPoly3		
	d	Result	Time (s)	d, k	Result	Time (s)	Result	Time (s)	Order, optimal
	3	−5.8852e−001	3.5						
	4	−4.2500e−001	508.8						
`C8_4`	≤ 3	Infeasible		1, 2	−6.9574e−001	3.3	−6.9574e−001	1.2	2, yes
	4	−1.1094e+000	1167.3						
`C10_2`	1	Infeasible		1, 1	−2.5000e−001	0.8	−2.5000e−001	0.3	1, yes
	2	−9.0000e−001	0.1						
	3	−5.8852e−001	15.8						
	4	−4.2500e−001	9993.0						
`C10_4`	≤ 3	Infeasible		1, 2	−6.9574e−001	11.4	−6.9574e−001	5.5	2, yes
	4	−1.1094e+000	5544.2						
`C20_2`	1	Infeasible		1, 1	−2.5000e−001	1.7	−2.5000e−001	0.4	1, yes
	2	−9.0000e−001	0.9						
	3	−5.8852e−001	2398.0						

Table 3 Comparison of BSOS and GloptiPoly3 on quadratic problems with polyhedral constraints

Problem	BSOS				GloptiPoly3			
	(d, k)	Result	Time (s)	Rank($\mathbf{M}_\ell(\mathbf{y}^*)$)	Order	Result	Time (s)	optimal
Qn10_r2	1, 1	Infeasible	0.8		1	Infeasible	0.1	
$n = 10, r = 2$	2,1	$-2.8023e{+}000$	0.5	1	2	$-2.8023e{+}000$	2.8	Yes
Qn10_r5	1, 1	Infeasible	0.7		1	Infeasible	0.1	
$n = 10, r = 5$	2,1	$-1.9685e{+}001$	0.4	1	2	$-1.9685e{+}001$	2.3	Yes
Qn20_r2	1, 1	Infeasible	1.5		1	Infeasible	0.1	
$n = 20, r = 2$	2, 1	$-2.0356e{-}003$	1.9	1	2	$-2.0356e{-}003$	4057.0	Yes
Qn20_r5	1, 1	Infeasible	1.7		1	Infeasible	0.1	
$n = 20, r = 5$	2, 1	$-1.7900e{+}001$	1.0	1	2	$-1.7900e{+}001$	3587.4	Yes
Qn40_r4	1, 1	Infeasible	10.9		1	Infeasible	1.2	
$n = 40, r = 4$	2, 1	$-7.0062e{+}000$	10.9	1	2	Out of memory		
Qn50_r5	1, 1	Infeasible	24.9		1	Infeasible	1.5	
$n = 50, r = 5$	2, 1	$-5.9870e{+}000$	34.8	1	2	Out of memory		
Qn100_r10	1, 1	Infeasible	385.8		1	Infeasible	108.7	
$n = 100, r = 10$	2, 1	$-8.8502e{+}000$	1617.4	1	2	Out of memory		
Problem 2.9 in Floudas and Pardalos (1990)	1, 1	Infeasible	0.9		1	Infeasible	0.1	
$n = 10, r = 6$	2, 1	3.7500e-001	0.5	1	2	$3.7500e - 001$	2.7	Yes
A = −toeplitz([0,1,1,1,1,zeros(1,10)])	1, 1	Infeasible	1.2		1	Infeasible	0.1	
$n = 15, r = 3$	2, 1	$-8.0000e{-}001$	0.6	11	2	$-8.0087e{-}001$	233.7	Unknown
A = − toeplitz([0,1,1,zeros(1,17)])	1, 1	Infeasible	1.9		1	Infeasible	0.2	
$n = 20, r = 7$	2, 1	$-6.6667e{-}001$	1.3	18	2	$-6.6905e{-}001$	4753.3	Unknown

Table 4 Comparison of BSOS and GloptiPoly3 on higher order problems with more variables

Problem	BSOS				GloptiPoly3			
	(d, k)	Result	Time (s)	Rank($\mathbf{M}_\ell(\mathbf{y}^*)$)	Order	Result	Time (s)	Optimal
`Hn20_2`	1, 1	−2.7002e+000	1.1	2	1	−2.7002e+000	2.3	Unknown
$n = 20, \ell = 2$	2, 1	−2.3638e+000	1.5	1	2	Out of memory		
`Hn20_4`	1, 2	−2.1860e+000	254	>10	1	Out of memory		
$n = 20, \ell = 4$	2, 2	−1.5943e+000	294	1				
`Hn30_2`	1, 1	−2.1320e+000	2.0	2	1	−2.1320e+000	2.5	Unknown
$n = 30, \ell = 2$	2, 1	−1.8917e+000	12	1	2	Out of memory		
`Hn30_4`	1, 2	−3.1126e+000	418	>10	1	Out of memory		
$n = 30, \ell = 4$	2, 2	−1.0781e+000	512	1				
`Hn40_2`	1, 1	−2.3789e+000	4.6	2	1	−2.3789e+000	4.0	Unknown
$n = 40, \ell = 2$	2, 1	−2.1138e+000	29	1	2	Out of memory		
`Hn40_4`	1, 2	−3.2917e+000	812	>10	1	Out of memory		
$n = 40, \ell = 4$	2, 2	−1.6531e+000	1102	1				

$$\begin{array}{ll} \min & \sum_{|\alpha|\leq \ell} c_\alpha \, x^\alpha \\ \text{s.t.} & x_i \geq 0, \quad i = 1, \ldots n \\ & \sum_{i=1}^{n} x_i^2 \leq 1 \end{array} \tag{12}$$

where $\ell = 2$ or $\ell = 4$ and c_α are randomly generated in $[-1, 1]$. In our numerical experiments, we generated instances such as `Hn20_4` for which $n = 20$ and the degree of the polynomial is 4. The problem is find the minimal value of a polynomial on the Euclidean unit ball intersected with the positive orthant.

Table 4 displays the respective performance of BSOS and GloptiPoly3. From the numerical results, we can see that BSOS is far more efficient than GloptiPoly3 in solving problems (12). For example, for problem `Hn20_4` (with $n = 20$ variables and degree $\ell = 4$), BSOS took about 250s to generate the first lower bound -2.1860, and took nearly 300s to generate a better lower bound of -1.5943, which is also the exact optimal value for the problem. But GloptiPoly3 got out of memory when solving the same problem. Similarly for problem `Hn40_2`, it took BSOS and GloptiPoly3 very little time to generate the first lower bound of -2.3789. To get a better lower bound, it took BSOS 29s to generate the optimal value of the problem. In contrast GloptiPoly3 got out of memory for improving the bound. From our observations, the disparity in efficiency between BSOS and GloptiPoly will become wider for instances with larger n and/or of higher degree.

5 Conclusion

We have described and tested a new hierarchy of semidefinite relaxations for global polynomial optimization. It tries to combine some advantages of previously defined LP- and SOS-hierarchies. Essentially, it uses a positivity certificate already used in the LP-hierarchy but with an additional semidefinite constraint which thus makes it an SOS-hierarchy. However the main and crucial point is that the size of this additional semidefinite constraint is fixed in advance and decided by the user (in contrast to the standard SOS-hierarchy in which the size of the semidefinite constraint increases in the hierarchy). Preliminary results are encouraging especially for non convex problems on convex polytopes where problems with up to 100 variables have been solved in a reasonable amount of time (whereas the standard SOS-hierarchy of GloptiPoly cannot be implemented).

For problems of larger size one needs to consider some serious numerical issues due to the presence of some fully dense submatrix and some nearly dependent linear constraints. In addition, to be able to handle large-scale problems one also needs to provide a "sparse version" of this hierarchy, an analogue of the sparse version of the SOS-hierarchy defined in Waki et al. (2006). Both issues (a topic of further investigation) are certainly non trivial, in particular the latter issue because the positivity certificate used in this new hierarchy involves products of initial polynomial constraints, which destroys the sparsity pattern considered in Waki et al. (2006).

Acknowledgements The work of the first author is partially supported by a PGMO grant from *Fondation Mathématique Jacques Hadamard, and an ERC-ADG grant from the European Research Council (ERC): grant agreement 666981 TAMING.*

Appendix

Before proving Lemma 1 we need introduce some notation. Given $k \in \mathbb{N}$ fixed, let $\tau = \max\{\deg(f), 2k, d \max_j\{\deg(g_j)\}\}$. For a sequence $\mathbf{y} = (y_\alpha) \in \mathbb{N}^n_\tau$, let $L_{\mathbf{y}} : \mathbb{R}[x]_\tau \to \mathbb{R}$ be the Riesz functional:

$$f \left(:= \sum_{\alpha \in \mathbb{N}^n_\tau} f_\alpha \, x^\alpha \right) \quad \mapsto \quad L_{\mathbf{y}}(f) := \sum_{\alpha \in \mathbb{N}^n_\tau} f_\alpha \, y_\alpha, \qquad f \in \mathbb{R}[x]_\tau,$$

and let $\mathbf{M}_k(\mathbf{y})$ be the moment matrix of order k, associated with $\mathbf{y}$. If $q \in \mathbb{R}[x]_k$ with coefficient vector $\mathbf{q} = (q_\alpha)$, then $\langle \mathbf{q}, \mathbf{M}_k(\mathbf{y})\, \mathbf{q} \rangle = L_{\mathbf{y}}(q^2)$ and if $\mathbf{y}$ is the (truncated) moment sequence of a measure μ,

$$\langle \mathbf{q}, \mathbf{M}_k(\mathbf{y})\, \mathbf{q} \rangle = L_{\mathbf{y}}(q^2) = \int q(x)^2 \, \mathrm{d}\mu(x).$$

Proof of Lemma 1

(a) We first prove that the dual of (7) which is the semidefinite program:

$$\rho^k_d := \inf_{\mathbf{y} \in \mathbb{R}^L} \{ L_{\mathbf{y}}(f) \, : \, \mathbf{M}_k(\mathbf{y}) \succeq 0; \; L_{\mathbf{y}}(1) = 1; \quad L_{\mathbf{y}}(h_{\alpha\beta}) \geq 0, \quad (\alpha, \beta) \in \mathbb{N}^{2m}_d \} \tag{13}$$

satisfies Slater's condition. Recall that K has nonempty interior; so let $\mathbf{y}$ be the sequence of moments of the Lebesgue measure μ on $\mathbf{K}$, scaled to be a probability measure, so that $L_{\mathbf{y}}(1) = 1$. Necessarily $\mathbf{M}_k(\mathbf{y}) \succ 0$. Otherwise there would exists $0 \neq q \in \mathbb{R}[x]_k$ such that

$$\langle \mathbf{q}, \mathbf{M}_k(\mathbf{y})\, \mathbf{q} \rangle = \int_K q(x)^2 \, \mathrm{d}\mu(x) = 0.$$

But then q vanishes almost everywhere on K, which implies $q = 0$, a contradiction.

Next, observe that for each $(\alpha, \beta) \in \mathbb{N}^{2m}_d$, the polynomial $h_{\alpha\beta} \in \mathbb{R}[x]_\tau$ is nonnegative on K and since there exists $x_0 \in K$ such that $0 < g_j(x_0) < 1$ for all $j = 1, \ldots, m$, there is an open set $O \subset K$ such that $h_{\alpha\beta}(x) > 0$ on O for all $(\alpha, \beta) \in \mathbb{N}^{2m}$. Therefore

$$L_{\mathbf{y}}(h_{\alpha\beta}) = \int_K h_{\alpha\beta} \, \mathrm{d}\mu \; \geq \; \int_O h_{\alpha\beta} \, \mathrm{d}\mu \; > 0, \quad \forall \, (\alpha, \beta) \in \mathbb{N}^{2m}.$$

Therefore $\mathbf{y}$ is a strictly feasible solution of (13), that is, Slater's condition holds true for (13). Hence $\rho^k_d = q^k_d$ for all d. It remains to prove that $q^k_d > -\infty$. But this follows from Theorem 1(b) as soon as d is sufficiently large, say $d \geq d_0$ for some integer d_0. Indeed then $-\infty < \theta_d \leq q^k_d \leq f^*$ for all $d \geq d_0$ (where θ_d is defined in (4)). Finally for each fixed d, (7) and (8) have same optimal value q_d^k and an optimal solution (q_d^k, λ^*, Q^*) of (7) is also an optimal solution of (8).

(b) Let θ^* be an optimal solution of (9) and let $\mathbf{y}^*$ be as in (10).

• If rank $\mathbf{M}_s(\mathbf{y}^*) = 1$ then $\mathbf{M}_s(\mathbf{y}^*) = v_s(x^*)\, v_s(x^*)^T$ for some $x^* \in \mathbb{R}^n$; this is due to the Hankel-like structure of the moment matrix combined with the rank-one property. So by definition of the moment matrix $\mathbf{M}_s(\mathbf{y}^*)$, $\mathbf{y}^* = (y_\alpha^*)$, $\alpha \in \mathbb{N}^n_{2s}$, is the vector of moments (up to order $2s$) of the Dirac measure δ_{x^*} at the point x^*. That is, $y_\alpha^* = (x^*)^\alpha$ for every $\alpha \in \mathbb{N}^n_{2s}$. But from (10),

$$(x^*)^\alpha = y_\alpha^* = \sum_{p=1}^{L} \theta_p^* \, (x^{(p)})^\alpha, \quad \forall \alpha \in \mathbb{N}^n_{2s}.$$

In particular, for moments of order 1 we obtain $x^* = \sum_{p=1}^L \theta_p^* \, x^{(p)}$. In other words, up to moments of order $2s$, one cannot distinguish the Dirac measure δ_{x^*} at x^* from the signed measure $\mu = \sum_p \theta_p^* \delta_{x^{(p)}}$ (recall that the θ_p^*'s are not necessarily nonnegative). That is, $(x^*)^\alpha = \int x^\alpha d\delta_{x^*} = \int x^\alpha \mathrm{d}\mu$ for all $\alpha \in \mathbb{N}^n_{2s}$. This in turn implies that for every $q \in \mathbb{R}[x]_{2s}$:

$$q(x^*) = \langle q, \delta_{x^*} \rangle = \langle q, \mu \rangle = \left\langle q, \sum_{p=1}^{L} \theta_p^* \, \delta_{x^{(p)}} \right\rangle = \sum_{p=1}^{L} \theta_p^* \, q(x^{(p)}).$$

Next, as θ^* is feasible for (9) and $2s \geq \max[\deg(f);\ \deg(g_j)]$,

$$0 \leq \sum_{p=1}^{L} \theta_p^* \, \langle h_{\alpha\beta}, \delta_{x^{(p)}} \rangle = \left\langle h_{\alpha\beta}, \sum_{p=1}^{L} \theta_p^* \, \delta_{x^{(p)}} \right\rangle = h_{\alpha\beta}(x^*), \quad \forall (\alpha, \beta) :\ \deg(h_{\alpha\beta}) \leq 2s.$$

In particular, choosing $(\alpha, \beta) \in \mathbb{N}^{2m}_{2s}$ such that $h_{\alpha\beta} = g_j$ (i.e. $\beta = 0$, $\alpha_i = \delta_{i=j}$), one obtains $g_j(x^*) \geq 0$, $j = 1, \ldots, m$, which shows that $x^* \in K$. In addition,

$$f^* \ \geq \ \tilde{q}_d^k = \sum_{p=1}^{L} \theta_p^* \, \langle f, \delta_{x^{(p)}} \rangle = \left\langle f, \sum_{p=1}^{L} \theta_p^* \, \delta_{x^{(p)}} \right\rangle = f(x^*),$$

which proves that $x^* \in K$ is an optimal solution of problem (P).

• If $\mathbf{M}_s(\mathbf{y}^*) \succeq 0, \mathbf{M}_{s-r}(g_j\, \mathbf{y}^*) \succeq 0, j = 1, \ldots, m,$ and rank $\mathbf{M}_s(\mathbf{y}^*) =$ rank $\mathbf{M}_{s-r}(\mathbf{y}^*)$ then by Curto and Fialkow (2000, 2005, Theorem 1.1), $\mathbf{y}^*$ is the vector of moments up to order $2s$, of some atomic-probability measure μ supported on $v :=$ rank $\mathbf{M}_s(\mathbf{y}^*)$ points $z(i) \in K$, $i = 1, \ldots, v$. That is, there exist positive weights $(w_i) \subset \mathbb{R}_+$ such that

$$\mu = \sum_{i=1}^{v} w_i \, \delta_{z(i)}; \quad \sum_{i=1}^{v} w_i = 1; \quad w_i \ > \ 0, \ i = 1, \ldots, v.$$

Therefore,

$$f^* \geq \tilde{q}_d^k = \sum_{p=1}^{L} \theta_p^* \langle f, \delta_{x(p)} \rangle = \sum_{\alpha \in \mathbb{N}^n} f_\alpha \, y_\alpha^* = \int_K f \, \mathrm{d}\mu \geq f^*,$$

which shows that $\tilde{q}_d^k = f^*$. In addition

$$0 = f^* - \int_K f \, \mathrm{d}\mu = \int_K (f^* - f) \, \mathrm{d}\mu = \sum_{i=1}^{v} \underbrace{w_i}_{>0} \underbrace{(f^* - f(z(i))}_{\leq 0},$$

which implies $f(z(i)) = f^*$ for every $i = 1, \ldots, v$. Finally, the v global minimizers can be extracted from the moment matrix $\mathbf{M}_s(\mathbf{y}^*)$ by the simple linear algebra procedure described in Henrion and Lasserre (2005). □

Test functions for BSOS and GloptiPoly in Table 1

Example `P4_2` (4 variables, degree 2):

$$\begin{aligned}
f &= x_1^2 - x_2^2 + x_3^2 - x_4^2 + x_1 - x_2;\\
g_1 &= 2x_1^2 + 3x_2^2 + 2x_1x_2 + 2x_3^2 + 3x_4^2 + 2x_3x_4;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4;\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4;\\
g_4 &= x_1^2 + 4x_2^2 - 3x_1x_2 + x_3^2 + 4x_4^2 - 3x_3x_4;\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4; \quad x \geq 0.
\end{aligned}$$

Example `P4_4` (4 variables, degree 4):

$$\begin{aligned}
f &= x_1^4 - x_2^4 + x_3^4 - x_4^4;\\
g_1 &= 2x_1^4 + 3x_2^2 + 2x_1x_2 + 2x_3^4 + 3x_4^2 + 2x_3x_4;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4;\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4;\\
g_4 &= x_1^2 + 4x_2^4 - 3x_1x_2 + x_3^2 + 4x_4^4 - 3x_3x_4;\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4; \quad x \geq 0.
\end{aligned}$$

Example `P4_6` (4 variables, degree 6):

$$\begin{aligned}
f &= x_1^4x_2^2 + x_1^2x_2^4 - x_1^2x_2^2 + x_3^4x_4^2 + x_3^2x_4^4 - x_3^2x_4^2;\\
g_1 &= x_1^2 + x_2^2 + x_3^2 + x_4^2;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4;\\
g_3 &= x_1^2 + 6x_2^4 - 8x_1x_2 + x_3^2 + 6x_4^4 - 8x_3x_4 + 2.5;
\end{aligned}$$

$$g_4 = x_1^4 + 3x_2^4 + x_3^4 + 3x_4^4; \quad g_5 = x_1^2 + x_2^3 + x_3^2 + x_4^3; \quad x \geq 0.$$

Example `P4_8` (4 variables, degree 8):

$$\begin{aligned} f &= x_1^4x_2^2 + x_1^2x_2^6 - x_1^2x_2^2 + x_3^4x_4^2 + x_3^2x_4^6 - x_3^2x_4^2; \quad g_1 = x_1^2 + x_2^2 + x_3^2 + x_4^2;\\ g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4;\\ g_3 &= x_1^2 + 6x_2^4 - 8x_1x_2 + x_3^2 + 6x_4^4 - 8x_3x_4 + 2.5;\\ g_4 &= x_1^4 + 3x_2^4 + x_3^4 + 3x_4^4; \quad g_5 = x_1^2 + x_2^3 + x_3^2 + x_4^3; \quad x \geq 0. \end{aligned}$$

Example `P6_2` (6 variables, degree 2):

$$\begin{aligned} f &= x_1^2 - x_2^2 + x_3^2 - x_4^2 + x_5^2 - x_6^2 + x_1 - x_2;\\ g_1 &= 2x_1^2 + 3x_2^2 + 2x_1x_2 + 2x_3^2 + 3x_4^2 + 2x_3x_4 + 2x_5^2 + 3x_6^2 + 2x_5x_6;\\ g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2 + 2x_6^2 - 4x_5x_6;\\ g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6;\\ g_4 &= x_1^2 + 4x_2^2 - 3x_1x_2 + x_3^2 + 4x_4^2 - 3x_3x_4 + x_5^2 + 4x_6^2 - 3x_5x_6;\\ g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2 + 5x_6^2 + 3x_5x_6; \quad x \geq 0. \end{aligned}$$

Example `P6_4` (6 variables, degree 4):

$$\begin{aligned} f &= x_1^4 - x_2^2 + x_3^4 - x_4^2 + x_5^4 - x_6^2 + x_1 - x_2;\\ g_1 &= 2x_1^4 + x_2^2 + 2x_1x_2 + 2x_3^4 + x_4^2 + 2x_3x_4 + 2x_5^4 + x_6^2 + 2x_5x_6;\\ g_2 &= 3x_1^2 + x_2^2 - 4x_1x_2 + 3x_3^2 + x_4^2 - 4x_3x_4 + 3x_5^2 + x_6^2 - 4x_5x_6;\\ g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6;\\ g_4 &= x_1^2 + 3x_2^4 - 3x_1x_2 + x_3^2 + 3x_4^4 - 3x_3x_4 + x_5^2 + 3x_6^4 - 3x_5x_6;\\ g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2 + 5x_6^2 + 3x_5x_6, \quad x \geq 0. \end{aligned}$$

Example `P6_6` (6 variables, degree 6):

$$\begin{aligned} f &= x_1^6 - x_2^6 + x_3^6 - x_4^6 + x_5^6 - x_6^6 + x_1 - x_2;\\ g_1 &= 2x_1^6 + 3x_2^2 + 2x_1x_2 + 2x_3^6 + 3x_4^2 + 2x_3x_4 + 2x_5^6 + 3x_6^2 + 2x_5x_6;\\ g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2 + 2x_6^2 - 4x_5x_6;\\ g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6;\\ g_4 &= x_1^2 + 4x_2^6 - 3x_1x_2 + x_3^2 + 4x_4^6 - 3x_3x_4 + x_5^2 + 4x_6^6 - 3x_5x_6;\\ g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2 + 5x_6^2 + 3x_5x_6, \quad x \geq 0. \end{aligned}$$

Example `P6_8` (6 variables, degree 8):

$$\begin{aligned}
f &= x_1^8 - x_2^8 + x_3^8 - x_4^8 + x_5^8 - x_6^8 + x_1 - x_2;\\
g_1 &= 2x_1^8 + 3x_2^2 + 2x_1x_2 + 2x_3^8 + 3x_4^2 + 2x_3x_4 + 2x_5^8 + 3x_6^2 + 2x_5x_6;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2 + 2x_6^2 - 4x_5x_6;\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6;\\
g_4 &= x_1^2 + 4x_2^8 - 3x_1x_2 + x_3^2 + 4x_4^8 - 3x_3x_4 + x_5^2 + 4x_6^8 - 3x_5x_6;\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2 + 5x_6^2 + 3x_5x_6, \quad x \ge 0.
\end{aligned}$$

Example `P8_2` (8 variables, degree 2):

$$\begin{aligned}
f &= x_1^2 - x_2^2 + x_3^2 - x_4^2 + x_5^2 - x_6^2 + x_7^2 - x_8^2 + x_1 - x_2;\\
g_1 &= 2x_1^2 + 3x_2^2 + 2x_1x_2 + 2x_3^2 + 3x_4^2 + 2x_3x_4 + 2x_5^2\\
&\quad + 3x_6^2 + 2x_5x_6 + 2x_7^2 + 3x_8^2 + 2x_7x_8;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2\\
&\quad + 2x_6^2 - 4x_5x_6 + 3x_7^2 + 2x_8^2 - 4x_7x_8;\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2\\
&\quad + 6x_6^2 - 4x_5x_6 + x_7^2 + 6x_8^2 - 4x_7x_8;\\
g_4 &= x_1^2 + 4x_2^2 - 3x_1x_2 + x_3^2 + 4x_4^2 - 3x_3x_4 + x_5^2\\
&\quad + 4x_6^2 - 3x_5x_6 + x_7^2 + 4x_8^2 - 3x_7x_8;\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4\\
&\quad + 2x_5^2 + 5x_6^2 + 3x_5x_6 + 2x_7^2 + 5x_8^2 + 3x_7x_8; \quad x \ge 0.
\end{aligned}$$

Example `P8_4` (8 variables, degree 4):

$$\begin{aligned}
f &= x_1^4 - x_2^4 + x_3^4 - x_4^4 + x_5^4 - x_6^4 + x_7^4 - x_8^4 + x_1 - x_2;\\
g_1 &= 2x_1^4 + 3x_2^2 + 2x_1x_2 + 2x_3^4 + 3x_4^2 + 2x_3x_4 + 2x_5^4\\
&\quad + 3x_6^2 + 2x_5x_6 + 2x_7^4 + 3x_8^2 + 2x_7x_8;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2\\
&\quad + 2x_6^2 - 4x_5x_6 + 3x_7^2 + 2x_8^2 - 4x_7x_8;\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2\\
&\quad + 6x_6^2 - 4x_5x_6 + x_7^2 + 6x_8^2 - 4x_7x_8;\\
g_4 &= x_1^2 + 4x_2^4 - 3x_1x_2 + x_3^2 + 4x_4^4 - 3x_3x_4 + x_5^2\\
&\quad + 4x_6^4 - 3x_5x_6 + x_7^2 + 4x_8^4 - 3x_7x_8;\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2\\
&\quad + 5x_6^2 + 3x_5x_6 + 2x_7^2 + 5x_8^2 + 3x_7x_8, \quad x \ge 0.
\end{aligned}$$

Example P8_6 (8 variables, degree 6):

$$\begin{aligned}
f &= x_1^6 - x_2^6 + x_3^6 - x_4^6 + x_5^6 - x_6^6 + x_7^6 - x_8^6 + x_1 - x_2;\\
g_1 &= 2x_1^6 + 3x_2^2 + 2x_1x_2 + 2x_3^6 + 3x_4^2 + 2x_3x_4 + 2x_5^6\\
&\quad + 3x_6^2 + 2x_5x_6 + 2x_7^6 + 3x_8^2 + 2x_7x_8;\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2\\
&\quad + 2x_6^2 - 4x_5x_6 + 3x_7^2 + 2x_8^2 - 4x_7x_8;\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2\\
&\quad + 6x_6^2 - 4x_5x_6 + x_7^2 + 6x_8^2 - 4x_7x_8;\\
g_4 &= x_1^2 + 4x_2^6 - 3x_1x_2 + x_3^2 + 4x_4^6 - 3x_3x_4 + x_5^2\\
&\quad + 4x_6^6 - 3x_5x_6 + x_7^2 + 4x_8^6 - 3x_7x_8;\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2\\
&\quad + 5x_6^2 + 3x_5x_6 + 2x_7^2 + 5x_8^2 + 3x_7x_8, \quad x \geq 0.
\end{aligned}$$

Example P10_2 (10 variables, degree 2):

$$\begin{aligned}
f &= x_1^2 - x_2^2 + x_3^2 - x_4^2 + x_5^2 - x_6^2 + x_7^2 - x_8^2 + x_9^2 - x_{10}^2 + x_1 - x_2;\\
g_1 &= 2x_1^2 + 3x_2^2 + 2x_1x_2 + 2x_3^2 + 3x_4^2 + 2x_3x_4 + 2x_5^2 + 3x_6^2 + 2x_5x_6\\
&\quad + 2x_7^2 + 3x_8^2 + 2x_7x_8 + 2x_9^2 + 3x_{10}^2 + 2x_9x_{10};\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2 + 2x_6^2 - 4x_5x_6\\
&\quad + 3x_7^2 + 2x_8^2 - 4x_7x_8 + 3x_9^2 + 2x_{10}^2 - 4x_9x_{10};\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6\\
&\quad + x_7^2 + 6x_8^2 - 4x_7x_8 + x_9^2 + 6x_{10}^2 - 4x_9x_{10};\\
g_4 &= x_1^2 + 4x_2^2 - 3x_1x_2 + x_3^2 + 4x_4^2 - 3x_3x_4 + x_5^2 + 4x_6^2 - 3x_5x_6\\
&\quad + x_7^2 + 4x_8^2 - 3x_7x_8 + x_9^2 + 4x_{10}^2 - 3x_9x_{10};\\
g_5 &= 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2 + 5x_6^2 + 3x_5x_6\\
&\quad + 2x_7^2 + 5x_8^2 + 3x_7x_8 + 2x_9^2 + 5x_{10}^2 + 3x_9x_{10}; \quad x \geq 0.
\end{aligned}$$

Example P10_4 (10 variables, degree 4):

$$\begin{aligned}
f &= x_1^4 - x_2^4 + x_3^4 - x_4^4 + x_5^4 - x_6^4 + x_7^4 - x_8^4 + x_9^4 - x_{10}^4 + x_1 - x_2;\\
g_1 &= 2x_1^4 + 3x_2^2 + 2x_1x_2 + 2x_3^4 + 3x_4^2 + 2x_3x_4 + 2x_5^4 + 3x_6^2 + 2x_5x_6\\
&\quad + 2x_7^4 + 3x_8^2 + 2x_7x_8 + 2x_9^4 + 3x_{11}^2 + 2x_9x_{10};\\
g_2 &= 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2 + 2x_6^2 - 4x_5x_6\\
&\quad + 3x_7^2 + 2x_8^2 - 4x_7x_8 + 3x_9^2 + 2x_{10}^2 - 4x_9x_{10};\\
g_3 &= x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6\\
&\quad + x_7^2 + 6x_8^2 - 4x_7x_8 + x_9^2 + 6x_{10}^2 - 4x_9x_{10};\\
g_4 &= x_1^2 + 4x_2^4 - 3x_1x_2 + x_3^2 + 4x_4^4 - 3x_3x_4 + x_5^2 + 4x_6^4 - 3x_5x_6
\end{aligned}$$

$$
\begin{aligned}
&+x_7^2+4x_8^4-3x_7x_8+x_9^2+4x_{10}^4-3x_9x_{10};\\
g_5 &= 2x_1^2+5x_2^2+3x_1x_2+2x_3^2+5x_4^2+3x_3x_4+2x_5^2+5x_6^2+3x_5x_6\\
&+2x_7^2+5x_8^2+3x_7x_8+2x_9^2+5x_{10}^2+3x_9x_{10}; \quad x \geq 0.
\end{aligned}
$$

Example P20_2 (20 variables, degree 2):

$$
\begin{aligned}
f &= x_1^2-x_2^2+x_3^2-x_4^2+x_5^2-x_6^2+x_7^2-x_8^2+x_9^2-x_{10}^2+x_{11}^2-x_{12}^2+x_1-x_2\\
&+x_{13}^2-x_{14}^2+x_{15}^2-x_{16}^2+x_{17}^2-x_{18}^2+x_{19}^2-x_{20}^2;\\
g_1 &= 2x_1^2+3x_2^2+2x_1x_2+2x_3^2+3x_4^2+2x_3x_4+2x_5^2+3x_6^2+2x_5x_6+2x_7^2+3x_8^2\\
&+2x_7x_8+2x_9^2+3x_{10}^2+2x_9x_{10}+2x_{11}^2+3x_{12}^2+2x_{11}x_{12}+2x_{13}^2+3x_{14}^2\\
&+2x_{13}x_{14}+2x_{15}^2+3x_{16}^2+2x_{15}x_{16}+2x_{17}^2+3x_{18}^2+2x_{17}x_{18}+2x_{19}^2+3x_{10}^2\\
&+2x_{20}x_{20};\\
g_2 &= 3x_1^2+2x_2^2-4x_1x_2+3x_3^2+2x_4^2-4x_3x_4+3x_5^2+2x_6^2-4x_5x_6+3x_7^2+2x_8^2\\
&-4x_7x_8+3x_9^2+2x_{10}^2-4x_9x_{10}+3x_{11}^2+2x_{12}^2-4x_{11}x_{12}+3x_{13}^2+2x_{14}^2\\
&-4x_{13}x_{14}+3x_{15}^2+2x_{16}^2-4x_{15}x_{16}+3x_{17}^2+2x_{19}^2-4x_{18}x_{18}+3x_{19}^2+2x_{20}^2\\
&-4x_{19}x_{20};\\
g_3 &= x_1^2+6x_2^2-4x_1x_2+x_3^2+6x_4^2-4x_3x_4+x_5^2+6x_6^2-4x_5x_6+x_7^2+6x_8^2-4x_7x_8\\
&+x_9^2+6x_{10}^2-4x_9x_{10}+x_{11}^2+6x_{12}^2-4x_{11}x_{12}+x_{13}^2+6x_{14}^2-4x_{13}x_{14}\\
&+x_{15}^2+6x_{17}^2-4x_{16}x_{16}+x_{17}^2+6x_{18}^2-4x_{17}x_{18}+x_{19}^2+6x_{20}^2-4x_{19}x_{20};\\
g_4 &= x_1^2+4x_2^2-3x_1x_2+x_3^2+4x_4^2-3x_3x_4+x_5^2+4x_6^2-3x_5x_6+x_7^2+4x_8^2-3x_7x_8\\
&+x_9^2+4x_{10}^2-3x_9x_{10}+x_1^2+4x_{12}^2-3x_{11}x_{12}+x_{13}^2+4x_{14}^2-3x_{15}x_{14}\\
&+x_{15}^2+4x_{16}^2-3x_{15}x_{16}+x_{17}^2+4x_{18}^2-3x_{17}x_{18}+x_{19}^2+4x_{20}^2-3x_{19}x_{20};\\
g_5 &= 2x_1^2+5x_2^2+3x_1x_2+2x_3^2+5x_4^2+3x_3x_4+2x_5^2+5x_6^2+3x_5x_6+2x_7^2+5x_8^2\\
&+3x_7x_8+2x_9^2+5x_{10}^2+3x_9x_{10}+2x_{11}^2+5x_{13}^2+3x_{12}x_{12}+2x_{13}^2+5x_{14}^2\\
&+3x_{13}x_{14}+2x_{15}^2+5x_{16}^2+3x_{15}x_{16}+2x_{17}^2+5x_{18}^2+3x_{17}x_{18}+2x_{19}^2+5x_{20}^2\\
&+3x_{19}x_{20}; \quad x \geq 0.
\end{aligned}
$$

Example P20_4 (20 variables, degree 4): same as P20_2 except that f is replaced by

$$
\begin{aligned}
f &= x_1^4-x_2^4+x_3^2-x_4^2+x_5^2-x_6^2+x_7^2-x_8^2+x_9^2-x_{10}^2+x_{11}^2-x_{12}^2+x_1-x_2\\
&+x_{13}^2-x_{14}^2+x_{15}^2-x_{16}^2+x_{17}^2-x_{18}^2+x_{19}^2-x_{20}^2;
\end{aligned}
$$

Test functions for BSOS versus LP relaxations of Krivine-Stengle on convex problems in Table 2

Example C4_2 (4 variables, degree 2):

$$
\begin{aligned}
f &= x_1^2+x_2^2+x_3^2+x_4^2+2x_1x_2-x_1-x_2;\\
g_1 &= -x_1^2-2x_2^2-x_3^2-2x_4^2+1;
\end{aligned}
$$

$$\begin{aligned}
g_2 &= -2x_1^2 - x_2^2 - 2x_3^2 - x_4^2 + 1;\\
g_3 &= -x_1^2 - 4x_2^2 - x_3^2 - 4x_4^2 + 1.25;\\
g_4 &= -4x_1^2 - x_2^2 - 4x_3^2 - x_4^2 + 1.25;\\
g_5 &= -2x_1^2 - 3x_2^2 - 2x_3^2 - 3x_4^2 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C4_4` (4 variables, degree 4):

$$\begin{aligned}
f &= x_1^4 + x_2^4 + x_3^4 + x_4^4 + 3x_1^2x_2^2 - x_1 - x_2;\\
g_1 &= -x_1^4 - 2x_2^4 - x_3^4 - 2x_4^4 + 1;\\
g_2 &= -2x_1^4 - x_2^4 - 2x_3^4 - x_4^4 + 1;\\
g_3 &= -x_1^4 - 4x_2^4 - x_3^4 - 4x_4^4 + 1.25;\\
g_4 &= -4x_1^4 - x_2^4 - 4x_3^4 - x_4^4 + 1.25;\\
g_5 &= -2x_1^4 - 3x_2^2 - 2x_3^4 - 3x_4^2 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C4_6` (4 variables, degree 6):

$$\begin{aligned}
f &= x_1^6 + x_2^6 + x_3^6 + x_4^6 + \frac{10}{3}x_1^3x_2^3 - x_1 - x_2;\\
g_1 &= -x_1^6 - 2x_2^6 - x_3^6 - 2x_4^6 + 1;\\
g_2 &= -2x_1^6 - x_2^6 - 2x_3^6 - x_4^6 + 1;\\
g_3 &= -x_1^6 - 4x_2^2 - x_3^6 - 4x_4^2 + 1.25;\\
g_4 &= -4x_1^6 - x_2^2 - 4x_3^6 - x_4^2 + 1.25;\\
g_5 &= -2x_1^2 - 3x_2^6 - 2x_3^2 - 3x_4^6 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C6_2` (6 variables, degree 2):

$$\begin{aligned}
f &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + 2x_1x_2 - x_1 - x_2;\\
g_1 &= -x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 - x_5^2 - 2x_6^2 + 1;\\
g_2 &= -2x_1^2 - x_2^2 - 2x_3^2 - x_4^2 - 2x_5^2 - x_6^2 + 1;\\
g_3 &= -x_1^2 - 4x_2^2 - x_3^2 - 4x_4^2 - x_5^2 - 4x_6^2 + 1.25;\\
g_4 &= -4x_1^2 - x_2^2 - 4x_3^2 - x_4^2 - 4x_5^2 - x_6^2 + 1.25;\\
g_5 &= -2x_1^2 - 3x_2^2 - 2x_3^2 - 3x_4^2 - 2x_5^2 - 3x_6^2 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C6_4` (6 variables, degree 4):

$$\begin{aligned}
f &= x_1^4 + x_2^4 + x_3^4 + x_4^4 + x_5^4 + x_6^4 + 3x_1^2x_2^2 - x_1 - x_2;\\
g_1 &= -x_1^4 - 2x_2^4 - x_3^4 - 2x_4^4 - x_5^4 - 2x_6^4 + 1;\\
g_2 &= -2x_1^4 - x_2^4 - 2x_3^4 - x_4^4 - 2x_5^4 - x_6^4 + 1;\\
g_3 &= -x_1^4 - 4x_2^4 - x_3^4 - 4x_4^4 - x_5^4 - 4x_6^4 + 1.25;
\end{aligned}$$

$$\begin{aligned}
g_4 &= -4x_1^4 - x_2^4 - 4x_3^4 - x_4^4 - 4x_5^4 - x_6^4 + 1.25;\\
g_5 &= -2x_1^4 - 3x_2^2 - 2x_3^4 - 3x_4^2 - 2x_5^4 - 3x_6^2 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C6_6` (6 variables, degree 6):

$$\begin{aligned}
f &= x_1^6 + x_2^6 + x_3^6 + x_4^6 + x_5^6 + x_6^6 + \frac{10}{3}x_1^2x_2^3 - x_1 - x_2;\\
g_1 &= -x_1^6 - 2x_2^6 - x_3^6 - 2x_4^6 - x_5^6 - 2x_6^6 + 1;\\
g_2 &= -2x_1^6 - x_2^6 - 2x_3^6 - x_4^6 - 2x_5^6 - x_6^6 + 1;\\
g_3 &= -x_1^6 - 4x_2^2 - x_3^6 - 4x_4^2 - x_5^6 - 4x_6^2 + 1.25;\\
g_4 &= -4x_1^6 - x_2^2 - 4x_3^6 - x_4^2 - 4x_5^6 - x_6^2 + 1.25;\\
g_5 &= -2x_1^2 - 3x_2^6 - 2x_3^2 - 3x_4^6 - 2x_5^2 - 3x_6^6 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C8_2` (8 variables, degree 2):

$$\begin{aligned}
f &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2 + 2x_1x_2 - x_1 - x_2;\\
g_1 &= -x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 - x_5^2 - 2x_6^2 - x_7^2 - 2x_8^2 + 1;\\
g_2 &= -2x_1^2 - x_2^2 - 2x_3^2 - x_4^2 - 2x_5^2 - x_6^2 - 2x_7^2 - x_8^2 + 1;\\
g_3 &= -x_1^2 - 4x_2^2 - x_3^2 - 4x_4^2 - x_5^2 - 4x_6^2 - x_7^2 - 4x_8^2 + 1.25;\\
g_4 &= -4x_1^2 - x_2^2 - 4x_3^2 - x_4^2 - 4x_5^2 - x_6^2 - 4x_7^2 - x_8^2 + 1.25;\\
g_5 &= -2x_1^2 - 3x_2^2 - 2x_3^2 - 3x_4^2 - 2x_5^2 - 3x_6^2 - 2x_7^2 - 3x_8^2 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C8_4` (8 variables, degree 4):

$$\begin{aligned}
f &= x_1^4 + x_2^4 + x_3^4 + x_4^4 + x_5^4 + x_6^4 + x_7^4 + x_8^4 + 3x_1^2x_2^2 - x_1 - x_2;\\
g_1 &= -x_1^4 - 2x_2^4 - x_3^4 - 2x_4^4 - x_5^2 - 2x_6^4 - x_7^4 - 2x_8^4 + 1;\\
g_2 &= -2x_1^4 - x_2^4 - 2x_3^4 - x_4^4 - 2x_5^2 - x_6^4 - 2x_7^4 - x_8^4 + 1;\\
g_3 &= -x_1^4 - 4x_2^4 - x_3^4 - 4x_4^4 - x_5^4 - 4x_6^4 - x_7^4 - 4x_8^4 + 1.25;\\
g_4 &= -4x_1^4 - x_2^4 - 4x_3^4 - x_4^4 - 4x_5^4 - x_6^4 - 4x_7^4 - x_8^4 + 1.25;\\
g_5 &= -2x_1^4 - 3x_2^2 - 2x_3^4 - 3x_4^2 - 2x_5^4 - 3x_6^2 - 2x_7^4 - 3x_8^2 + 1.1; \quad x \geq 0.
\end{aligned}$$

Example `C10_2` (10 variables, degree 2):

$$\begin{aligned}
f &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2 + x_9^2 + x_{10}^2 + 2x_1x_2 - x_1 - x_2;\\
g_1 &= -x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 - x_5^2 - 2x_6^2 - x_7^2 - 2x_8^2 - x_9^2 - 2x_{10}^2 + 1;\\
g_2 &= -2x_1^2 - x_2^2 - 2x_3^2 - x_4^2 - 2x_5^2 - x_6^2 - 2x_7^2 - x_8^2 - 2x_9^2 - x_{10}^2 + 1;\\
g_3 &= -x_1^2 - 4x_2^2 - x_3^2 - 4x_4^2 - x_5^2 - 4x_6^2 - x_7^2 - 4x_8^2 - x_9^2 - 4x_{10}^2 + 1.25;
\end{aligned}$$

$$\begin{aligned} g_4 &= -4x_1^2 - x_2^2 - 4x_3^2 - x_4^2 - 4x_5^2 - x_6^2 - 4x_7^2 - x_8^2 - 4x_9^2 - x_{10}^2 + 1.25;\\ g_5 &= -2x_1^2 - 3x_2^2 - 2x_3^2 - 3x_4^2 - 2x_5^2 - 3x_6^2 - 2x_7^2 - 3x_8^2 - 2x_9^2\\ &\quad -3x_{10}^2 + 1.1; \quad x \geq 0. \end{aligned}$$

Example `C10_4` (10 variables, degree 4):

$$\begin{aligned} f &= x_1^4 + x_2^4 + x_3^4 + x_4^4 + x_5^4 + x_6^4 + x_7^4 + x_8^4 + x_9^4 + x_{10}^4 + 3x_1^2x_2^2 - x_1 - x_2;\\ g_1 &= -x_1^4 - 2x_2^4 - x_3^4 - 2x_4^4 - x_5^4 - 2x_6^4 - x_7^4 - 2x_8^4 - x_9^4 - 2x_{10}^4 + 1;\\ g_2 &= -2x_1^4 - x_2^4 - 2x_3^4 - x_4^4 - 2x_5^4 - x_6^4 - 2x_7^4 - x_8^4 - 2x_9^4 - x_{10}^4 + 1;\\ g_3 &= -x_1^4 - 4x_2^4 - x_3^4 - 4x_4^4 - x_5^4 - 4x_6^4 - x_7^4 - 4x_8^4 - x_9^4 - 4x_{10}^4 + 1.25;\\ g_4 &= -4x_1^4 - x_2^4 - 4x_3^4 - x_4^4 - 4x_5^4 - x_6^4 - 4x_7^4 - x_8^4 - 4x_9^4 - x_{10}^4 + 1.25;\\ g_5 &= -2x_1^4 - 3x_2^2 - 2x_3^4 - 3x_4^2 - 2x_5^4 - 3x_6^2 - 2x_7^4 - 3x_8^2 - 2x_9^4\\ &\quad -3x_{10}^2 + 1.1; \quad x \geq 0. \end{aligned}$$

Example `C20_2` (20 variables, degree 2):

$$\begin{aligned} f &= x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2 + x_9^2 + x_{10}^2 + 2x_1x_2 - x_1 - x_2\\ &\quad + x_{11}^2 + x_{12}^2 + x_{13}^2 + x_{14}^2 + x_{15}^2 + x_{16}^2 + x_{17}^2 + x_{18}^2 + x_{19}^2 + x_{20}^2;\\ g_1 &= -x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 - x_5^2 - 2x_6^2 - x_7^2 - 2x_8^2 - x_9^2 - 2x_{10}^2\\ &\quad - x_{11}^2 - 2x_{12}^2 - x_{13}^2 - 2x_{14}^2 - x_{15}^2 - 2x_{16}^2 - x_{17}^2 - 2x_{18}^2 - x_{19}^2 - 2x_{20}^2 + 1;\\ g_2 &= -2x_1^2 - x_2^2 - 2x_3^2 - x_4^2 - 2x_5^2 - x_6^2 - 2x_7^2 - x_8^2 - 2x_9^2 - x_{10}^2\\ &\quad - 2x_{11}^2 - x_{12}^2 - 2x_{13}^2 - x_{14}^2 - 2x_{15}^2 - x_{16}^2 - 2x_{17}^2 - x_{18}^2 - 2x_{19}^2 - x_{20}^2 + 1;\\ g_3 &= -x_1^2 - 4x_2^2 - x_3^2 - 4x_4^2 - x_5^2 - 4x_6^2 - x_7^2 - 4x_8^2 - x_9^2 - 4x_{10}^2\\ &\quad - x_{11}^2 - 4x_{12}^2 - x_{13}^2 - 4x_{14}^2 - x_{15}^2 - 4x_{16}^2 - x_{17}^2 - 4x_{18}^2 - x_{19}^2 - 4x_{20}^2 + 1.25;\\ g_4 &= -4x_1^2 - x_2^2 - 4x_3^2 - x_4^2 - 4x_5^2 - x_6^2 - 4x_7^2 - x_8^2 - 4x_9^2 - x_{10}^2\\ &\quad - 4x_{11}^2 - x_{12}^2 - 4x_{13}^2 - x_{14}^2 - 4x_{15}^2 - x_{16}^2 - 4x_{17}^2 - x_{18}^2 - 4x_{19}^2 - x_{20}^2 + 1.25;\\ g_5 &= -2x_1^2 - 3x_2^2 - 2x_3^2 - 3x_4^2 - 2x_5^2 - 3x_6^2 - 2x_7^2 - 3x_8^2 - 2x_9^2 - 3x_{10}^2\\ &\quad - 2x_{11}^2 - 3x_{12}^2 - 2x_{13}^2 - 3x_{14}^2 - 2x_{15}^2 - 3x_{16}^2 - 2x_{17}^2 - 3x_{18}^2 - 2x_{19}^2\\ &\quad - 3x_{20}^2 + 1.1; \quad x \geq 0. \end{aligned}$$

References

Ahmadi AA, Majumdar A (2014) DSOS and SDSOS Optimization: LP and SOCP-Based Alternatives to SOS Optimization. In: Proceedings of the 48th annual conference on information sciences and systems, Princeton, NJ, USA, pp 1–5

Benabbas S, Magen A (2010) Extending SDP integrality gaps to Sherali–Adams with applications to quadratic programming and MaxCutGain, In: Shepherd FB (ed) Integer programming and combinatorial optimization. Lecture Notes in Computer Science, Springer, New York, pp 299–312

Curto RE, Fialkow LA (2000) The truncated complex K-moment problem. Trans Amer Math Soc 352:2825–2855

Curto RE, Fialkow LA (2005) Truncated K-moment problem in several variables. J Op Theory 54:189–226
Chlamtac E, Tulsiani M (2012) Convex relaxations and integrality gaps. In: Anjos M, Lasserre JB (eds) Handbook of semidefinite, conic and polynomial optimization. Springer, New York, pp 139–170
de Klerk E, Laurent M (2011) On the Lasserre hierarchy of semidefinite programming relaxations of convex polynomial optimization problems. SIAM J Optim 21:824–832
Floudas CA, Pardalos PM (1990) A collection of test problems for constrained global optimization algorithms. Lecture Notes in Computer Science, vol 455, Springer, Berlin
Henrion D, Lasserre JB (2005) Detecting global optimality and extracting solutions in Gloptipoly. In: Henrion D, Garulli A(eds) Positive polynomials in control, Lecture Notes in Control and Information Science, vol 312, pp 293–310
Henrion D, Lasserre JB, Lofberg J (2009) GloptiPoly 3: moments, optimization and semidefinite programming. Optim Methods Softw 24:761–779
Helton JW, Nie J (2010) Semidefinite representations of convex sets. Math Program 122:21–64
Krivine JL (1964) Anneaux préordonnés. J Anal Math 12:307–326
Lasserre JB (2001) Global optimization with polynomials and the problem of moments. SIAM J Optim 11:796–817
Lasserre JB (2002) Semidefinite programming vs. LP relaxations for polynomial programming. Math Op Res 27:347–360
Lasserre JB (2009) Convexity in semi-algebraic geometry and polynomial optimization. SIAM J Optim 19:1995–2014
Lasserre JB (2009) Moments, positive polynomials and their applications. Imperial College Press, London
Lasserre JB (2013) A Lagrangian relaxation view of linear and semidefinite hierarchies. SIAM J Optim 23:1742–1756
Laurent M (2003) A comparison of the Sherali–Adams, Lovász-Schrijver and Lasserre relaxations for 0–1 programming. Math Op Res 28:470–496
Marshall M (2009) Representation of non-negative polynomials, degree bounds and applications to optimization. Can J Math 61:205–221
Nie J (2014) Optimality conditions and finite convergence of Lasserre's hierarchy. Math Program 146:97–121
Putinar M (1993) Positive polynomials on compact semi-algebraic sets. Ind Univ Math J 42:969–984
Sherali HD, Adams WP (1990) A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. SIAM J Discr Math 3:411–430
Sherali HD, Adams WP (1999) A reformulation-linearization technique for solving discrete and continuous nonconvex problems. Kluwer, Dordrecht, MA
Stengle G (1974) A nullstellensatz and a positivstellensatz in semialgebraic geometry. Math Ann 207:87–97
Toh KC, Todd MJ, Tutuncu RH (1999) SDPT3—a Matlab software package for semidefinite programming. Optim Methods Softw 11:545–581
Toh KC, Todd MJ, Tutuncu RH (2003) Solving semidefinite-quadratic-linear programs using SDPT3. Math Program 95:189–217
Waki S, Kim S, Kojima M, Maramatsu M (2006) Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity. SIAM J Optim 17:218–242