UNIVERSITY OF CALIFORNIA, SAN DIEGO

# Semidefinite Relaxations Approach to Polynomial Optimization and Its Extensions

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Li Wang

Committee in charge:

Professor Jiawang Nie, Chair
Professor J. William Helton, Co-Chair
Professor Philip E. Gill
Professor Gert Lanckriet
Professor Sonia Martinez Diaz
Professor Mauricio de Oliveira

2014

The dissertation of Li Wang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____
Co-Chair

_____
Chair

University of California, San Diego

2014

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

At the very beginning of my Ph.D. thesis, I would like to express my gratitude to all people that helped me to finish my Ph.D. study.

First and foremost, I would like to express my deepest thanks to my academic advisor Professor Jiawang Nie. He provided me with the great opportunity to work with the cutting-edge global optimization technique and gave me a lot of help to start this thesis. I am really fortunate to study under his guidance.

My grateful thanks to my academic co-advisor, Professor J. William Helton for his continuous support and his patience in explaining some interesting engineering problems, which gave me great help in understanding research.

I would also like to thank Postdoctoral Feng Guo, who gave me great help in studying polynomial optimization. It is a wonderful experience to study together with him, and we have successfully finished two research papers.

Special thanks to my parents and brother and sister in law. They gave me support and encouragement, which helped me to finish my study at UCSD.

Chapter 1 subsection 1.3.2, in part, and Chapter 2 Sections 2.1 and 2.3, in full, are reprint of the material as it appears in the article "Regularization Methods for SDP relaxations in Large Scale Polynomial Optimization" by Jiawang Nie and Li Wang, in SIAM Journal on Optimization, Volume 22, No.2(2012). The dissertation author was one of the authors of this paper.

Chapter 2 Section 2.2 and Chapter 3, in full, are reprint of the material as it appears in the article "Minimizing Rational Functions by Exact Jacobian SDP relaxation Applicable to Finite Singularities" by Feng Guo, Li Wang and Guangming Zhou, in the Journal of Global Optimization in volume 58, No.2(2014). The dissertation author was one of the authors of this paper.

Chapter 4, in full, is a reprint of the material as it appears in the article "Semidefinite Relaxations for Semi-Infinite Polynomial Programming" by Li Wang and Feng Guo, in Computational Optimization and Applications, Volume 58, No.1(2014). The dissertation author was the first author of this paper.

Chapter 5, in full, is a reprint of the material that has been submitted for publication as it may appear in the article "Semidefinite Relaxations on Tensor

Best Rank-1 Approximation" by Jiawang Nie and Li Wang, in SIAM Journal on Matrix Analysis and Applications, 2014. The dissertation author was one of the authors of this paper.

VITA

| | |
|---|---|
| 2002-2006 | B. S. in Computational Mathematics, China University of Mining and Technology, P.R.China |
| 2006-2009 | M. S. in Computational Mathematics, Xi'an Jiaotong University, P.R.China |
| 2009-2014 | Ph. D. in Mathematics, University of California, San Diego |

PUBLICATIONS

1) Jiawang Nie, **Li Wang**. Semidefinite Relaxations on Tensor Best Rank-1 Approximation. Submitted to: *SIAM Journal on Matrix Analysis and Applications*. Revised, 2014.

2) **Li Wang**, Feng Guo. Semidefinite Relaxations for Semi-Infinite Polynomial Programming. *Computational Optimization and Applications*, 58(1):133-159, 2014.

3) Jiawang Nie, **Li Wang**. Regularization Methods for SDP Relaxations in Large Scale Polynomial Optimization. *SIAM Journal on Optimization*, 22(2):408-428, 2012.

4) Feng Guo, **Li Wang**, Guangming Zhou. Minimizing Rational Functions by Exact Jacobian SDP Relaxation Applicable to Finite Singularities. *Journal of Global Optimization*, 58(2):261-284, 2014.

ABSTRACT OF THE DISSERTATION

Semidefinite Relaxations Approach to Polynomial Optimization and Its Extensions

by

Li Wang

Doctor of Philosophy in Mathematics

University of California, San Diego, 2014

Professor Jiawang Nie, Chair
Professor J. William Helton, Co-Chair

The goal of this thesis is to study a special nonlinear programming, namely, polynomial optimization in which both the objective and constraints are polynomials. This kind of problem is always NP-hard even if the objective is nonconvex quadratic and all constraints are linear. The semidefinite (SDP) relaxations approach, based on sum of squares representations, provides us with strong tools to solve polynomial optimization problems with finitely many constraints globally.

We first review two SDP relaxation methods for solving polynomial optimization problems with finitely many constraints: the classic Lasserre's SDP relaxation and Jacobian SDP relaxation. In general, these methods relax the poly-

nomial optimization problem as a sequence of SDPs whose optima are the lower bounds of the global minimum and converge to the global minimum under certain assumptions. We also prove that the assumption of nonsingularity in Jacobian SDP relaxation method can be weakened to have finite singularities.

Then, we study the problem of minimizing a rational function. We reformulate the problem by the technique of homogenization, the original problem and the reformulated problem are shown to be equivalent under some generic conditions. The constraint set of the reformulated problem may not be compact, and Lasserre's SDP relaxation may not have finite convergence, so we apply Jacobian SDP relaxation to solve the reformulated polynomial optimization problem. Some numerical examples are presented to show the efficiency of this method.

Next, we consider the problem of minimizing semi-infinite polynomial programming (SIPP). We propose an exchange algorithm with SDP relaxations to solve SIPP problems with a compact index set globally. And we extend the proposed method to SIPP problems with noncompact index set via homogenization. The reformulated problem is equivalent to original SIPP problem under some generic conditions.

At last, we study the problem of finding best rank-1 approximations for both symmetric and nonsymmetric tensor. For symmetric tensors, this is equivalent to optimizing homogeneous polynomials over unit spheres; for nonsymmetric tensors, this is equivalent to optimizing multi-quadratic forms over multi-spheres. We use semidefinite relaxations approach to solve these polynomial optimization problems. Extensive numerical experiments are presented to show that this approach is practical in getting best rank-1 approximations.

# Chapter 1

# Introduction to Polynomial Optimization

Polynomial optimization is a specific nonlinear programming, in which the objective and constraints are all polynomials. The general formulation of polynomial optimization that we will concern with is:

$$\begin{cases} f_{\min} := \min\limits_{x\in\mathbb{R}^n} f(x) \\ \qquad\qquad \text{s.t. } h_i(x) = 0, \quad i \in \{1, \cdots, m_1\}, \\ \qquad\qquad\quad g_j(x) \geq 0, \quad j \in \{1, \cdots, m_2\}, \end{cases} \tag{1.1}$$

where $f(x), h_i(x), g_j(x) \in \mathbb{R}[x]$, and $\mathbb{R}[x] := \mathbb{R}[x_1, \cdots, x_n]$ denotes the ring of multivariate polynomials in the $n$-tuple of variables $(x_1, \cdots, x_n)$ with real coefficients. $m_1, m_2$ are integers. Let $K$ be the feasible region of problem (1.1). Problem (1.1) is generally nonconvex and NP-hard even when the objective function is nonconvex quadratic and the constraints are linear [59]. If $K = \mathbb{R}^n$, problem (1.1) is reduced to general unconstrained polynomial optimization problem. If $m_1 = \infty$ or $m_2 = \infty$, problem (1.1) is called semi-infinite polynomial programming (SIPP).

In this thesis, we firstly focus on solving problem (1.1) by semidefinite (SDP) relaxations methods, which will be presented in detail in Chapter 2. Then we discuss how to apply the SDP relaxation methods to solve three kinds of problems: (1) Minimizing rational functions; (2) Semi-infinite polynomial programming; (3) Best rank-1 tensor approximation problems.

## 1.1 Notation

The symbol $\mathbb{N}$ (resp., $\mathbb{R}$) denotes the set of nonnegative integers (resp., real numbers). For any $t \in \mathbb{R}$, $\lceil t \rceil$ denotes the smallest integer no less than $t$. For $x \in \mathbb{R}^n$, $x_i$ denotes the $i$-th component of $x$, that is, $x = (x_1, \ldots, x_n)$. $\mathbb{S}^{n-1}$ denotes the $n-1$ dimensional unit sphere $\{x \in \mathbb{R}^n : x_1^2 + \cdots + x_n^2 = 1\}$. For $\alpha \in \mathbb{N}^n$, denote $|\alpha| = \alpha_1 + \cdots + \alpha_n$. The symbol $\mathbb{N}_{\leq k}$ denotes the set $\{\alpha \in \mathbb{N}^n : |\alpha| \leq k\}$, and $\mathbb{N}_k$ denotes $\{\alpha \in \mathbb{N}^n : |\alpha| = k\}$. For each $i$, $e_i$ denotes the $i$-th standard unit vector. The $\mathbf{1}$ denotes a vector of all ones. For a finite set $T$, $|T|$ denotes its cardinality. For $x \in \mathbb{R}^n$ and $\alpha \in \mathbb{N}^n$, $x^\alpha$ denotes $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$. For a matrix $A$, $A^T$ denotes its transpose. The $I_N$ denotes the $N \times N$ identity matrix, and $\mathcal{S}_+^N$ denotes the cone of symmetric positive semidefinite $N \times N$ matrices. For any vector $u \in \mathbb{R}^N$, $\|u\| = \sqrt{u^T u}$ denotes the standard Euclidean norm. For integer $n > 0$, $[n]$ denotes the set $\{1, \cdots, n\}$. For a symmetric matrix $W$, $W \succeq 0 (\succ 0)$ means that $W$ is positive semidefinite (definite). The symbol $\Sigma_{n,m}$ denotes the cone of sum of squares forms in $n$ variables and of degree $m$. For two tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$, define their inner product as

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{1 \leq i_1 \leq n_1, \ldots, 1 \leq i_m \leq n_m} \mathcal{X}_{i_1,\ldots,i_m} \mathcal{Y}_{i_1,\ldots,i_m}.$$

If $m = 2$, this is the general matrix inner product.

## 1.2 Definitions and Preliminaries

In this section, we will introduce some basic definitions and results from linear algebra and algebra geometry needed for later discussion.

### 1.2.1 Definitions

**Definition 1.2.1.** (Nonnegative Polynomials)
*Given a polynomial $f(x) \in \mathbb{R}[x]$. We say $f(x)$ is positive semidefinite (PSD) or nonnegative over $K$, if the evaluation $f(x) \geq 0$ for every $x \in K$.*

**Definition 1.2.2.** (Sum of Squares)

*A polynomial is Sum of Squares* (SOS) *if it is a sum of squares of other polynomials, i.e., for given polynomial* $f \in \mathbb{R}[x]$*, there exists* $q_j \in \mathbb{R}[x]$*, such that*

$$f = \sum_{j=1}^{r} q_j^2.$$

Obviously, if a polynomial $f(x)$ is SOS, then $f(x)$ is positive semidefinite or nonnegative.

**Definition 1.2.3.** (Semialgebraic Set)
*In real algebraic geometry, a semialgebraic set of* $\mathbb{R}^n$ *is defined to be a set of the form:*

$$S = \{x \in \mathbb{R}^n : h_1(x) = \cdots = h_{m_1}(x) = 0, g_1(x) \geq 0, \cdots, g_{m_2}(x) \geq 0\}, \qquad (1.2)$$

*where* $h_i, g_j \in \mathbb{R}[x]$.

**Definition 1.2.4.** (Quadratic Module)
*Let* $S$ *be the semialgebraic set defined by polynomials* $h_i, g_j \in \mathbb{R}[x]$*. The quadratic module generated by the defining polynomials of* $S$ *is the set*

$$M(S) = \left\{ \sum_{i=1}^{m_1} \varphi_i h_i + \sum_{j=0}^{m_2} \sigma_j g_j : \ \textit{each } \sigma_j \textit{ is SOS} \right\}.$$

*Here* $g_0 = 1$.

**Definition 1.2.5.** (Preordering)
*Let* $S$ *be the semialgebraic set defined by polynomials* $h_i, g_j \in \mathbb{R}[x]$*. The preordering generated by the defining polynomials of* $S$ *is the set*

$$P(S) = \left\{ \sum_{i=1}^{m_1} \varphi_i h_i + \sum_{\nu \in \{0,1\}^{m_2}} \sigma_\nu g_1^{\nu_1} \cdots g_{m_2}^{\nu_{m_2}} : \textit{each } \sigma_\nu \textit{ is SOS} \right\}.$$

*Here* $g^0 = 1$.

**Definition 1.2.6.** (Ideal and Variety)
*A subset* $I$ *of* $\mathbb{R}[x]$ *is an ideal if* $p \cdot h \in I$ *for any* $p \in I$ *and* $h \in \mathbb{R}[x]$*. The variety of an ideal* $I$ *is the set of all common complex zeros of the polynomials in I, i.e.*

$$V(I) = \{x \in \mathbb{C}^n : \ p(x) = 0 \textit{ for all } p \in I\}.$$

If $g_1, \cdots, g_m \in \mathbb{R}[x]$, then $\langle g_1, \cdots, g_m \rangle$ denotes the smallest ideal containing the $g_i$, which is the set of all polynomials that are polynomial linear combination of the $g_i$, i.e.

$$\langle g_1, \cdots, g_m \rangle = \sum_{i=1}^{m} h_i g_i, \quad \forall\ h_i \in \mathbb{R}[x], i \in [m].$$

**Definition 1.2.7.** (Radical Ideal)
*Given any ideal $I \in \mathbb{R}[x]$, its radical is the ideal*

$$\sqrt{I} = \{q \in \mathbb{R}[x] :\ q^m \in I \text{ for some } m \in \mathbb{N}\}.$$

*Then $I \subseteq \sqrt{I}$, and we say ideal $I$ is radical if $I = \sqrt{I}$.*

**Definition 1.2.8.** (Cone and Dual Cone)
*A set $C$ is a cone if $ax \in C$ for all $a > 0$ and $x \in C$. The dual cone of $C$ is the set:*

$$C^* = \{y : \langle y, x \rangle \geq 0, \quad \forall x \in C\}.$$

**Definition 1.2.9.** (Flat Extension of Matrix)
*Let $X$ be a symmetric matrix with the block form*

$$X = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}.$$

*We say that $X$ is a flat extension of matrix $A$ if rank $X$ = rank $A$ or, equivalently, if $B = AW$ and $C = B^T W = W^T A W$ for some matrix $W$.*

### 1.2.2 Preliminaries on Polynomials

Denote the two sets of polynomials:

$$\begin{aligned} P_{n,d} &= \{f \in \mathbb{R}[x] :\ f(x) \text{ is PSD of degree } d\}, \\ \Sigma_{n,d} &= \{f \in \mathbb{R}[x] :\ f(x) \text{ is SOS of degree } d\}. \end{aligned} \tag{1.3}$$

**Theorem 1.2.10.** (Hilbert Theorem [11]). *$\Sigma_{n,d} \subseteq P_{n,d}$ and the equality holds if and only if (1) $n = 1$, or (2) $d = 2$, or (3) $(n, d)$ = (2,4).*

This theorem implies that if a univariate polynomial $f(x)$ is nonnegative, then $f(x)$ must be SOS. Actually it must be a sum of two squares [42]. Not all nonnegative polynomials is SOS. For examples [65],

- Motzkin polynomial:

$$M(x,y,z) = x^4y^2 + x^2y^4 + z^6 - 3x^2y^2z^2$$

- Robinson Polynomial:

$$R(x,y,z) = x^6+y^6+z^6-(x^4y^2+x^2y^4+x^4z^2+x^2z^4+y^4z^2+y^2z^4)+3x^2y^2z^2$$

- Choi-Lam:

$$\begin{aligned} F(x_1,x_2,x_3;y_1,y_2,y_3) = {} & x_1^2y_1^2 + x_2^2y_2^2 + x_3^2y_3^2 + 2(x_1^2y_2^2 + x_2^2y_3^2 + x_3^2y_1^2) \\ & - 2x_1x_2y_1y_2 - 2x_1x_3y_1y_3 - 2x_2x_3y_2y_3 \end{aligned}$$

**Theorem 1.2.11.** (Hilbert Basis Theorem [10])
*Every ideal $I \subset \mathbb{R}[x]$ has a finite generating set, i.e., $I = \langle g_1, \cdots, g_m \rangle$ for some $g_1, \cdots, g_m \in I$.*

**Theorem 1.2.12.** (Hilbert Weak Nullstellensatz [10])
*Given an ideal $I \in \mathbb{R}[x]$ with $V(I) = \emptyset$ , then $1 \in I$.*

**Theorem 1.2.13.** (Hilbert Strong Nullstellensatz [10])
*Given an ideal $I \in \mathbb{R}[x]$, then $I(V(I)) = \sqrt{I}$.*

**Theorem 1.2.14.** (Putinar's Positivstellensatz [63])
*Let $S$ be a compact semialgebraic set. Suppose there exists $R > 0$ such that*

$$R - \|x\|^2 \in M(S).$$

*If $f(x)$ is positive on $S$, then $f(x) \in M(S)$.*

**Theorem 1.2.15.** (Schmüdgen's Positivstellensatz [70])
*Let $S$ be a compact semialgebraic set. If $f(x)$ is positive on $S$, then $f(x) \in P(S)$.*

## 1.3 Semidefinite Programming

In this section, we review some terminologies and duality theory in Semidefinite Programming (SDP). SDP has been an active research area over the past two decades, which is motivated by its importance in both optimization theory and wide applications. We refer to the review paper [45] and the book [17] for theory and applications on SDP.

### 1.3.1 Theory

Consider the primal problem of standard SDP:

$$\begin{cases} p^* := \inf\limits_{X \in \mathcal{S}^N} & C \bullet X \\ \text{s.t.} & \mathcal{A}(X) = b,\ X \succeq 0. \end{cases} \tag{1.4}$$

Here $\mathcal{S}^N$ denotes the space of $N \times N$ real symmetric matrices, $X \succeq 0$ (resp. $X \succ 0$) means $X$ is positive semidefinite (resp. definite), and $\bullet$ denotes the standard Frobenius inner product, i.e., for given $A, B \in \mathcal{S}^N$, we have

$$A \bullet B = \langle A, B \rangle = tr(A^T B) = \sum_{i,j=1}^{N} a_{ij} b_{ij}.$$

The $C \in \mathcal{S}^N$ and $b \in \mathbb{R}^m$ are constant, and $\mathcal{A} : \mathcal{S}^N \to \mathbb{R}^m$ is a linear operator given by $\mathcal{A}(X) = (A_1 \bullet X, \cdots, A_m \bullet X)^T$, where $A_i \in \mathcal{S}^N,\ \forall\ i \in [m]$.

The dual problem of (1.4) is

$$\begin{cases} d^* := \sup\limits_{y \in \mathbb{R}^m, Z \in \mathcal{S}^N} & b^T y \\ \text{s.t.} & \mathcal{A}^*(y) + Z = C,\ Z \succeq 0. \end{cases} \tag{1.5}$$

Here $\mathcal{A}^*$ is the adjoint of $\mathcal{A}$ given by $\mathcal{A}^*(y) = \sum\limits_{i=1}^{m} y_i A_i$. Let $\mathcal{S}^N_+$ be the set of $N \times N$ positive semidefinite matrices, then $\mathcal{S}^N_+$ is a closed convex cone. Moreover, the cone $\mathcal{S}^N_+$ is self-dual, i.e.,

$$(\mathcal{S}^N_+)^* = \{Z \in \mathcal{S}^N :\ \langle X, Z \rangle \geq 0,\ \forall\ X \in \mathcal{S}^N_+\} = \mathcal{S}^N_+.$$

**Theorem 1.3.1.** (Weak Duality) *For any feasible $X$ for primal problem (1.4), and feasible $y$ for dual problem (1.5), it holds that $C \bullet X \geq b^\top y$. We have*

$$p^* \geq d^*. \tag{1.6}$$

*Proof.* It is easy to see that, for any feasible $X$ and $(y, Z)$ for problems (1.4) and (1.5) respectively, we have

$$C \bullet X - b^T y = X \bullet Z \geq 0.$$

Then the proof is completed. □

It is possible that $p^*$ or $d^*$ are not achievable, and the equality in (1.6) does not hold. For example:

**Example 1.3.2.**

(1) $\min \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \bullet X, \quad \text{s.t.} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \bullet X = -1, \quad \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \bullet X = 0, \quad X \succeq 0.$

Its dual problem is

$$\max \ -y_1 \ \text{ s.t. } \begin{bmatrix} y_1 & 1 \\ 1 & y_2 \end{bmatrix} \succeq 0.$$

It holds that $p^* = d^* = 0$, but $d^*$ is not achievable.

(2) $\min \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \bullet X, \quad \text{s.t.} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \bullet X = 0, \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \bullet X = 2, \quad X \succeq 0.$

This SDP problem is infeasible, so $p^* = +\infty$. Its dual problem is

$$\max \ 2y_2 \ \text{ s.t. } \begin{bmatrix} -y_1 & -y_2 \\ -y_2 & 0 \end{bmatrix} \succeq 0.$$

The optimal value $d^* = 0$, and we have $p^* > d^*$.

(3) $\min \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet X, \ \text{s.t.} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \bullet X = 0, \ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} \bullet X = 2, \ X \succeq 0.$

Any feasible solution $X$ for the above has the form

$$\begin{bmatrix} 0 & \xi_1 & \xi_2 \\ \xi_1 & \xi_3 & \xi_4 \\ \xi_2 & \xi_4 & 1 - \xi_1 \end{bmatrix} \succeq 0.$$

The optimal value is $p^* = 1$. Its dual problem is:

$$\max\ 2y_2\ \ \text{s.t.}\ \begin{bmatrix} -y_2 & -y_2 & 0 \\ -y_2 & 0 & 0 \\ 0 & 0 & 1-2y_2 \end{bmatrix} \succeq 0.$$

The optimal value is $d^* = 0$. Both $p^*$ and $d^*$ are achievable, but $p^* > d^*$.

**Theorem 1.3.3.** (Strong Duality [45, Theorem 3.1]) *If there exists $X \succ 0$ which is feasible for problem (1.4) and problem (1.5) has feasible solution, then $p^* = d^*$ and the supremum of problem (1.5) is achievable; Similarly, if there exists $y, Z \succ 0$ which is feasible for problem (1.5) and problem (1.4) has feasible solution, then $p^* = d^*$ and the infimum of problem (1.4) is achievable.*

For convenience, in the following, we assume that optima of the primal and dual problems (1.4) and (1.5) are achievable, and we use min and max instead of inf and sup.

SDPs are convex programs that are always regarded as an extension of linear programming (LP) where all the vector inequalities are replaced with matrix inequalities. The strong duality Theorem 1.3.3 implies the difference between SDPs and LPs. Recall that for LP, either the primal or the dual is feasible and bounded, then the primal and dual optimal values are equal and achievable. For SDPs, the results in Theorem 1.3.3 are weaker. However, the similarities between SDPs and LPs motivate us to generalize the efficient algorithms for LPs to solve SDPs. And following the success of the methods for large scale linear programming, many efficient algorithms have been proposed to solve SDPs. Among them, one of the most popular algorithms is the primal-dual central path following method [17], which tackles both the primal and dual problems simultaneously. Even though it has polynomial worst-case complexity [45], it is slow in solving very large scale SDPs. In subsection 2.3.1, we will show its difficulty in solving polynomial optimization problems.

### 1.3.2 Algorithms

In this subsection, we give a brief overview of primal-dual interior point method and regularization method for solving SDPs. Since the SDP relaxations for constrained polynomial optimization often have block diagonal structure, e.g., (2.11), we consider the standard SDPs with block diagonal structures.

Let $\mathcal{K}$ be a cross product of several semidefinite cones

$$\mathcal{K} = \mathcal{S}_+^{N_1} \times \cdots \times \mathcal{S}_+^{N_\ell}.$$

Consider the general conic semidefinite optimization problem:

$$(P): \begin{cases} \min & C \bullet X \\ \text{s.t.} & \mathcal{A}(X) = b,\ X \in \mathcal{K}. \end{cases} \tag{1.7}$$

Here $C \in \mathcal{M}$, $b \in \mathbb{R}^m$, and $\mathcal{A} : \mathcal{M} \to \mathbb{R}^m$ is a linear operator. The dual of (1.7) is

$$(D): \begin{cases} \max & b^T y \\ \text{s.t.} & \mathcal{A}^*(y) + Z = C,\ Z \in \mathcal{K}. \end{cases} \tag{1.8}$$

$\mathcal{K}$ belongs to the space $\mathcal{M} = \mathcal{S}^{N_1} \times \cdots \times \mathcal{S}^{N_\ell}$. Each $X \in \mathcal{M}$ is a tuple $X = (X_1, \ldots, X_\ell)$ with every $X_i \in \mathcal{S}^{N_i}$. So, $X$ could also be thought of as a symmetric block diagonal matrix, and $X \in \mathcal{K}$ if and only if its every block $X_i \succeq 0$. The notation $X \succeq_{\mathcal{K}} 0$ (resp. $X \succ_{\mathcal{K}} 0$) means every block of $X$ is positive semidefinite (resp. definite). For $X = (X_1, \ldots, X_\ell) \in \mathcal{M}$ and $Y = (Y_1, \ldots, Y_\ell) \in \mathcal{M}$, we define their inner product as $X \bullet Y = X_1 \bullet Y_1 + \cdots + X_\ell \bullet Y_\ell$. Denote by $\|\cdot\|$ the norm in $\mathcal{M}$ induced by this inner product. Similar to one block case, $\mathcal{K}$ is also a self-dual cone, that is,

$$\mathcal{K}^* = \{Y \in \mathcal{M} : Y \bullet X \geq 0, \quad \forall\ X \in \mathcal{K}\} = \mathcal{K}.$$

For a symmetric matrix $W$, denote by $(W)_+$ (resp. $(W)_-$) the projection of $W$ into the positive (resp. negative) semidefinite cone, that is, if $W$ has spectral decomposition

$$W = \sum_{\lambda_i > 0} \lambda_i u_i u_i^T + \sum_{\lambda_i < 0} \lambda_i u_i u_i^T,$$

then $(W)_+$ and $(W)_-$ are defined as

$$(W)_+ = \sum_{\lambda_i > 0} \lambda_i u_i u_i^T, \quad (W)_- = \sum_{\lambda_i < 0} \lambda_i u_i u_i^T.$$

For $X = (X_1, \ldots, X_\ell) \in \mathcal{M}$, its projections into $\mathcal{K}$ and $-\mathcal{K}$ are given by

$$(X)_{\mathcal{K}} = ((X_1)_+, \ldots, (X_\ell)_+), \quad (X)_{-\mathcal{K}} = ((X_1)_-, \ldots, (X_\ell)_-).$$

The optimality conditions for primal $(P)$ and dual $(D)$ problems are:

$$\begin{cases} \mathcal{A}(X) = b, \\ \mathcal{A}^*(y) + S = C, \\ XS = 0, \\ X \in \mathcal{K}, \ \ S \in \mathcal{K}. \end{cases} \tag{1.9}$$

To design efficient interior point method, we define the central path $XS = \mu I$, where $I$ is the block identity matrix with proper size and $\mu$ is a parameter. The basic iterations of primal-dual interior point method can be described as follows.

**Algorithm 1.3.4.** (Primal-Dual Interior Point Method [2])
***Input:*** *Choose a strictly feasible $X_0$ for $(P)$ and $(y_0, S_0)$ for $(D)$, $\mu > 0$, $\tau \in (0,1), \theta \in (0,1), \epsilon \in (0,1)$. Set $k = 0$.*
***Output:*** *Optimal solution $(X^*, y^*, Z^*)$ of primal dual problems (1.7)-(1.8).*
***Step 1****: Compute Newton's Direction $(\Delta X, \Delta y, \Delta S)$ from*

$$\begin{cases} \mathcal{A}(\Delta X) = b - \mathcal{A}(\Delta X_k) \\ \mathcal{A}^*(\Delta y) + \Delta S = C - \mathcal{A}^*(y_k) + S_k \\ \Delta X S_k + S_k \Delta X + X_k \Delta S + \Delta S X_k = 2\mu I - (X_k S_k + S_k X_k). \end{cases} \tag{1.10}$$

***Step 2****: Compute the step-length:*

$$\alpha_k = \max\{\alpha \in (0,1] : (X_k + \alpha \Delta X)_j \succ 0, \ (S_k + \alpha \Delta S)_j \succ 0, \ j \in [\ell]\}.$$

***Step 3****: Update $X_{k+1} = X_k + \alpha_k \Delta X$, $S_{k+1} = S_k + \alpha_k \Delta S$ and $y_{k+1} = y_k + \alpha \Delta y$.*
***Step 4****: Set $k = k + 1$. If $\|XS - (X_k \bullet S_k/n)I\|_F > \tau X_k \bullet S_k/n$, then go to Step 1. Otherwise, go to Step 5.*
***Step 5****: If $\mu > \epsilon$, update $\mu = \theta\mu$ and go to Step 1, otherwise, stop.*

Please refer to [2] for some variations and convergence properties of primal-dual interior point methods. Next, we consider the regularization method for conic

SDPs. They are natural generalizations of regularization methods introduced in [26, 27, 81] for solving standard SDP problems of one block structure.

There are two typical regularizations for SDP problems: *Moreau-Yosida regularization* for the primal (1.7) and *Augmented Lagrangian regularization* for the dual (1.8). They would be naturally generalized to conic semidefinite optimization (1.7) and its dual (1.8). The Moreau-Yosida regularization for (1.7) is

$$\begin{cases} \min\limits_{X,Y\in\mathcal{M}} & C\bullet X+\frac{1}{2\sigma}\|X-Y\|^2 \\ \text{s.t.} & \mathcal{A}(X)=b,\ X\in\mathcal{K}. \end{cases} \tag{1.11}$$

Obviously (1.11) is equivalent to (1.7), because for each fixed feasible $X\in\mathcal{M}$ the optimal $Y\in\mathcal{M}$ in (1.11) is equal to $X$. The Augmented Lagrangian regularization for (1.8) is

$$\begin{cases} \max\limits_{y\in\mathbb{R}^m,Z\in\mathcal{M}} & b^Ty-(Z+\mathcal{A}^*(y)-C)\bullet Y-\frac{\sigma}{2}\|Z+\mathcal{A}^*(y)-C\|^2 \\ \text{s.t.} & Z\in\mathcal{K}. \end{cases} \tag{1.12}$$

When $\mathcal{K}=\mathcal{S}_+^N$ is a single product, it was shown (see Section 2 of [26]) that for every fixed $Y$, (1.12) is the dual optimization problem of

$$\min_{X\in\mathcal{M}} \quad C\bullet X+\frac{1}{2\sigma}\|X-Y\|^2-y^T(\mathcal{A}(X)-b)-Z\bullet X.$$

By fixing $y\in\mathbb{R}^m$ and optimizing over $Z\succeq 0$, Malick, Povh, Rendl, and Wiegele [26] further showed that (1.12) can be reduced to

$$\max_{y\in\mathbb{R}^m} \quad b^Ty-\tfrac{\sigma}{2}\|(\mathcal{A}^*(y)-C+Y/\sigma)_{\mathcal{K}}\|^2+\tfrac{1}{2\sigma}\|Y\|^2. \tag{1.13}$$

When $\mathcal{K}=\mathcal{S}_+^N$ is a single product, Malick, Povh, Rendl, and Wiegele [26] proposed a general framework (see Algorithm 4.3 of [26]) of regularization methods for solving large scale SDP problems. It can be readily generalized to the case that $\mathcal{K}$ is a product of several semidefinite cones.

Denote by $\varphi_\sigma(Y,y)$ the objective in (1.13). When $\mathcal{K}=\mathcal{S}_+^N$, $\varphi_\sigma(Y,y)$ is differentiable [26, Proposition 3.2] and

$$\nabla_y\varphi_\sigma(Y,y)=b-\sigma\mathcal{A}\Big((\mathcal{A}^*(y)-C+Y/\sigma)_{\mathcal{K}}\Big).$$

The above is also true when $\mathcal{K}$ is a cross product of semidefinite cones. For fixed $Y_k$, we need to solve the following maximization problem

$$\max_{y\in\mathbb{R}^m} \quad \varphi_\sigma(Y_k, y). \tag{1.14}$$

Since $\varphi_\sigma$ is concave in $y$, a point $\hat{y}$ is a maximizer of (1.14) if and only if

$$\nabla_y \varphi_\sigma(Y_k, \hat{y}) = 0.$$

The function $\varphi_\sigma(Y, y)$ is not twice differentiable, so the standard Newton's method is not applicable. However, the function $\varphi_\sigma(Y, y)$ is semismooth, and semismooth Newton's method could be applied to get local superlinear or quadratic convergence, as pointed out in [81, Section 3.2]. Thus, the generalized Hessian of $\varphi_\sigma$ is required in computation. We refer to [81, Section 3.2] for a numerical method of evaluating $\nabla_y^2 \varphi_\sigma(Y, y)$. It is important to point out that the Hessian $\nabla_y^2 \varphi_\sigma(Y, y)$ does not need to be explicitly formulated. It would be implicitly available such that the matrix vector product $\nabla_y^2 \varphi_\sigma(Y, y) \cdot z$ would be evaluated efficiently.

Generally, we always have $\nabla_y^2 \varphi_\sigma(Y, y) \succeq 0$, and $\nabla_y^2 \varphi_\sigma(Y, y) \succ 0$ if some nondegeneracy conditions hold ([81, Proposition 3.2]). Therefore, an approximate semismooth Newton direction $d_{new}$ for (1.14) can be determined from the linear system

$$\Big(\nabla_y^2 \varphi_\sigma(Y, y) + \epsilon \cdot I_N\Big) d_{new} = \nabla_y \varphi_\sigma. \tag{1.15}$$

Here $\epsilon > 0$ is a tiny number ensuring the positive definiteness of the above linear system. When $m$ is huge, it is usually not practical to solve (1.15) by direct methods like Cholesky factorization. To avoid this difficulty, conjugate gradient (CG) iterations are suitable, as proposed in [81].

Now we describe the Newton-CG Augmented Lagrangian regularization method for solving (1.7)-(1.8) as follows.

**Algorithm 1.3.5.** (Newton-CG Regularization Method [81])
***Input:*** *Choose* $\epsilon^{in}, \epsilon^{out} \in (0, 1)$, $\epsilon > 0$, $\delta \in (0, 1)$, $\rho > 1$, $\sigma_{\max}, K \in \mathbb{N}$. *Choose* $X_0, Z_0 \in \mathcal{K}$, $y_0 \in \mathbb{R}^m$, $\sigma_0$ *and set* $k = 0$.
***Output:*** *Optimal solution* $(X^*, y^*, Z^*)$ *of primal dual problems (1.7)-(1.8).*
**Procedure:**

***Step 1***: *If* $\|Z_k + \mathcal{A}^*(y_k) - C\| < \epsilon^{out}$, *stop; otherwise, go to Step 2.*

***Step 2***: *Set* $Y_k := X_k$. *Set* $j = 0$ *and* $y_{k,j} = y_k$.

I *If* $\|\nabla_y \varphi_{\sigma_k}(Y_k, y_{k,j})\| < \epsilon^{in}$, *go to Step 3; otherwise, go to Step 2(II);*

II *Set* $g_j = \nabla_y \varphi_{\sigma_k}(Y_k, y_{k,j})$.

*Compute* $d_{new}$ *in (1.15) by applying preconditioned CG at most* $K$ *steps.*

*Find the smallest integer* $\alpha > 0$ *such that*

$$\varphi_{\sigma_k}(Y_k, y_{k,j} + \delta^\alpha \cdot d_{new}) \geq \varphi_{\sigma_k}(Y_k, y_{k,j}) + \delta^\alpha \cdot g_j^T d_{new}. \tag{1.16}$$

*Set* $y_{k,j+1} := y_{k,j} + \delta^\alpha \cdot d_{new}$.

*Compute the projections:*

$X_k = \sigma_k(Y_k/\sigma_k + \mathcal{A}^*(y_{k,j+1}) - C)_{\mathcal{K}}$, $Z_k = -(Y_k/\sigma_k + \mathcal{A}^*(y_{k,j+1}) - C)_{-\mathcal{K}}$.

*Set* $j := j + 1$.

***Step 3***: *Set* $y_{k+1} := y_{k,j}$. *If* $\sigma_k \leq \sigma_{\max}$, *set* $\sigma_{k+1} = \rho\sigma_k$.
*Set* $k := k + 1$, *go to Step 1.*

When $\mathcal{K} = \mathcal{S}_+^N$ is a single product, the convergence of Algorithm 1.3.5 has been discussed in [81, Theorems 3.5, 4.1, 4.2]. These results could be readily generalized to $\mathcal{K}$ being a product of several $\mathcal{S}_+^N$. We refer to [26, 66, 67, 81] for the convergence of Algorithm 1.3.5.

## 1.4 Outline of The Thesis

This thesis is organized as follows: In Chapter 2, we introduce two SDP relaxation methods for solving polynomial optimization problems with finitely many constraints: Lasserre's SDP relaxation (Section 2.1) and Jacobian SDP relaxation (Section 2.2). In Chapter 3, we discuss how to minimize rational functions by Jacobian SDP relaxations. We reformulate the problem as a polynomial optimization problem by the technique of homogenization. The two problems are shown to be equivalent under some generic conditions, and some numerical examples are

presented to support the superiority of our approach. In Chapter 4, we discuss how to solve semi-infinite polynomial programming (SIPP) problems by SDP relaxation methods. We first propose an exchange algorithm with SDP relaxations to solve SIPP problems with compact index set. Then we extend the proposed method to SIPP problems with noncompact index set via homogenization. Numerical results show that the algorithm is efficient in practice. In Chapter 5, we study the problem of finding best rank-1 approximations for both symmetric and nonsymmetric tensors. For symmetric tensors, this is equivalent to optimizing homogeneous polynomials over unit spheres; for nonsymmetric tensors, this is equivalent to optimizing multi-quadratic forms over multi-spheres. We propose semidefinite relaxations to solve these polynomial optimization problems. And extensive numerical experiments are presented to show that this approach is efficient in practice. We conclude this thesis in Chapter 6.

Chapter 1 subsection 1.3.2, in part, is a reprint of the material as it appears in the article "Regularization Methods for SDP relaxations in Large Scale Polynomial Optimization" by Jiawang Nie and Li Wang, in SIAM Journal on Optimization, Volume 22, No.2(2012). The dissertation author was one of the authors of this paper.

# Chapter 2

# Semidefinite Relaxations For Polynomial Optimization

In this Chapter, we study how to solve the following polynomial optimization problem with finitely many constraints:

$$\begin{cases} f_{\min} := \min\limits_{x\in\mathbb{R}^n} f(x) \\ \qquad\quad \text{s.t. } h_i(x) = 0, \quad i \in [m_1], \\ \qquad\qquad g_j(x) \geq 0, \quad j \in [m_2], \end{cases} \tag{2.1}$$

where $f(x), h_i(x), g_j(x) \in \mathbb{R}[x]$. Based on the Positivstellensatz, considerable works have recently been done on solving (2.1) by means of SDP relaxation. Generally speaking, these methods relax (2.1) as a sequence of SDPs whose optima are lower bounds of $f_{\min}$ and converge to $f_{\min}$ under some assumptions. We first introduce the classic Lasserre's SDP relaxation [39] and then Jacobian SDP relaxation [51] with the property of finite convergence.

## 2.1 Lasserre's SDP Relaxation

In this section, we review Lasserre's SDP relaxation [39] and show the construction of SDP relaxations for constrained polynomial optimization problems in detail as an example.

### 2.1.1 Algorithm Description

Denote $K$ as the feasible set of (2.1). Let $\mathcal{F} := \{h_1, \ldots, h_{m_1}, g_0, g_1, \ldots, g_{m_2}\}$ and $g_0 = 1$. The $k$-th truncated quadratic module generated by $\mathcal{F}$ is defined as

$$Q_k(\mathcal{F}) := \left\{ \sum_{j=1}^{m_1} \phi_j h_j + \sum_{i=0}^{m_2} \sigma_i g_i \,\middle|\, \begin{array}{l} \sigma_i \text{ are SOS},\ \phi_j \in \mathbb{R}[x],\ \forall\ i, j \\ \deg(\sigma_i g_i) \le 2k,\ \deg(\phi_j h_j) \le 2k \end{array} \right\}.$$

The $k$-th Lasserre's SDP relaxation [39] for solving (2.1) ($k$ is also called the relaxation order) is

$$f_k := \max\ \gamma \quad \text{s.t. } f(x) - \gamma \in Q_k(\mathcal{F}). \tag{2.2}$$

The relaxation (2.2) is equivalent to a semidefinite program and could be solved efficiently by numerical methods like interior-point method [45] and regularization method [81]. Clearly, $f_k \le f_{\min}$ for every $k$ and the sequence $\{f_k\}$ is monotonically increasing. The quadratic module generated by $\mathcal{F}$ is

$$Q(\mathcal{F}) := \bigcup_{k=1}^{\infty} Q_k(\mathcal{F}).$$

**Definition 2.1.1.** *The set $Q(\mathcal{F})$ satisfies the Archimedean Condition if there exists $\psi \in Q(\mathcal{F})$ such that inequality $\psi(x) \ge 0$ defines a compact set in $x \in \mathbb{R}^n$.*

Note that the Archimedean Condition implies the feasible set $K$ is compact but the inverse is not necessarily true. However, for any compact $K$ we can always "force" the associated quadratic module to satisfy the Archimedean Condition by adding a "redundant" constraint, e.g., $\rho - \|x\|^2 \ge 0$ for sufficiently large $\rho$.

The convergence for Lasserre's hierarchy (2.2), i.e., $\lim_{k\to\infty} f_k = f_{\min}$, is implied by Putinar's Positivstellensatz (Theorem 1.2.14):

**Theorem 2.1.2.** ([60]) *If a polynomial $p$ is positive on $K$ and the Archimedean Condition holds, then $p \in Q(\mathcal{F})$.*

We next consider the dual optimization problem of (2.2). Let $y$ be a truncated moment sequence (tms) of degree $2k$, i.e., $y = (y_\alpha)$ be a sequence of real numbers which are indexed by $\alpha := (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$ with $|\alpha| := \alpha_1 + \cdots + \alpha_n \le$

$2k$. The associated $k$-th moment matrix is denoted as $M_k(y)$ which is indexed by $\mathbb{N}^n_k$, with $(\alpha,\beta)$-th entry $y_{\alpha+\beta}$. Given polynomial $p(x) = \sum_\alpha p_\alpha x^\alpha$ where $x^\alpha := x_1^{\alpha_1}\cdots x_n^{\alpha_n}$, denote $d_p = \lceil \deg(p)/2 \rceil$. For $k \geq d_p$, the $(k-d_p)$-th localizing moment matrix $L_p^{(k-d_p)}(y)$ is defined as the moment matrix of the shifted vector $((py)_\alpha)_{\alpha\in\mathbb{N}^n_{2(k-d_p)}}$ with $(py)_\alpha = \sum_\beta p_\beta y_{\alpha+\beta}$. Denote by $\mathscr{M}_{2k}$ the space of all tms whose degrees are $2k$. Let $\mathbb{R}[x]_{2k}$ be the space of real polynomials in $x$ with degree at most $2k$. For any $y \in \mathscr{M}_{2k}$, a Riesz functional $\mathscr{L}_y$ on $\mathbb{R}[x]_{2k}$ is defined as

$$\mathscr{L}_y\left(\sum_\alpha q_\alpha x_1^{\alpha_1}\cdots x_n^{\alpha_n}\right) = \sum_\alpha q_\alpha y_\alpha, \quad \forall\ q(x) \in \mathbb{R}[x]_{2k}.$$

For convenience, we hereafter still use $q$ to denote the coefficient vector of $q(x)$ in the graded lexicographical ordering and denote $\langle q, y\rangle = \mathscr{L}_y(q)$. From the definition of the localizing moment matrix $L_p^{(k-d_p)}(y)$, it is easy to check that

$$q^T L_p^{(k-d_p)}(y) q = \mathscr{L}_y(p(x)q(x)^2), \quad \forall\ q(x) \in \mathbb{R}[x]_{k-d_p}.$$

The dual optimization problem of (2.2) is ([39, 40])

$$\begin{cases} f_k^* := \min\limits_{y\in\mathscr{M}_{2k}} \langle f, y\rangle \\ \qquad \text{s.t. } L_{h_j}^{(k-d_{h_j})}(y) = 0,\ j\in[m_1],\ L_{g_i}^{(k-d_{g_i})}(y) \succeq 0,\ i\in[m_2], \\ \qquad\quad M_k(y) \succeq 0,\ \langle 1, y\rangle = 1. \end{cases} \tag{2.3}$$

Let

$$d = \max\{1, d_{g_i}, d_{h_j} \mid i\in[m_1], j\in[m_2]\}.$$

Lasserre [39] shows that $f_k \leq f_k^* \leq f_{\min}$ for every $k \geq \max\{d_f, d\}$ and both $\{f_k\}$ and $\{f_k^*\}$ converge to $f_{\min}$ if the Archimedean Condition holds.

We say Lasserre's hierarchy (2.2) and (2.3) has *finite convergence* if

$$f_{k_1} = f_{k_1}^* = f_{\min} \quad \text{for some order } k_1 < \infty. \tag{2.4}$$

Interestingly, Nie proved that under the Archimedean Condition, Lasserre's SDP relaxation has finite convergence *generically* ([52, Theorem 1.1]). Since $f_{\min}$ is usually unknown, a practical issue is how to certify the finite convergence if it happens. Moreover, if it is certified, how do we get minimizers?

Let $y^*$ be an optimizer of (2.3). By [9, Theorem 1.1], $f_k^* = f_{\min}$ for some $k$ if the *flat extension condition* (FEC) [9] holds, i.e.,

$$\text{rank } M_{k-d}(y^*) = \text{rank } M_k(y^*). \tag{2.5}$$

By solving some SVD and eigenvalue problems ([18]), we can get $r := \text{rank } M_k(y^*)$ global optimizers for (2.1). In subsection 2.1.2, we will show the extraction process in details. However, (2.5) is not a generally necessary condition for checking finite convergence of Lasserre's hierarchy ([53, Example 1.1]). To certify the finite convergence of (2.2) and get minimizers of (2.1) from (2.3), a weaker condition was proposed in [53]. We say a minimizer $y^*$ of (2.3) satisfies *flat truncation condition* (FTC) if there exists an integer $t \in [\max\{d_f, d\}, k]$ such that

$$\text{rank } M_{t-d}(y^*) = \text{rank } M_t(y^*). \tag{2.6}$$

If an optimizer of (2.3) has a flat truncation, by [9, Theorem 1.1] again, we still have $f_k^* = f_{\min}$. Moreover, if there is no duality gap between (2.2) and (2.3), we obtain $f_k = f_{\min}$. More importantly, [53, Theorem 2.2] shows that the flat truncation is also necessary for Lasserre's hierarchy (2.2) under some *generic* assumptions. Furthermore, assuming the set of global minimizers is nonempty and finite, [53, Theorem 2.2 and 2.6] show that the Lasserre's hierarchy has finite convergence if and only if the flat truncation condition holds.

**Algorithm 2.1.3.** (Lasserre's SDP relaxation)
***Input:*** *Objective function $f(x)$, constraint functions $h_i(x), g_j(x)$ and maximal relaxation order $k_{\max}$.*
***Output:*** *Global minimum and minimizers of problem (2.1).*

I *Set $d := \max\{1, d_f, d_{h_i}, d_{g_j}\}$ and initial relaxation order $k = d$.*

II *Solve primal and dual SDP problems (2.2) and (2.3) by standard SDP solver (e.g., SeDuMi [74], SDPT3 [77], SDPNAL [83]).*

III *For $t \in [d, k]$, check condition (2.6).*

1 *If (2.6) holds for some $t$, get minimizers by Extraction Algorithm [18] and stop;*

2 *Otherwise, go to Step IV.*

IV *If $k > k_{\max}$, stop; otherwise, set $k = k + 1$ and go to Step II.*

### 2.1.2 Finite Convergence Certification

In this subsection, we present how to certify and find all the global minimizers if the finite convergence of Lasserre's SDP relaxation happens. As mentioned in above subsection, Lasserre's hierarchy has finite convergence if and only if the flat truncation condition (2.6) holds. If (2.6) holds, we are supposed to find $r := \text{rank } M_t(y^*)$ global minimizers. Next we review how to extract the solutions $x^*(j)$ by using the method described in [18].

Let $M_k(y^*)$ be the $k$th moment matrix. Since $M_k(y^*) \succeq 0$, its Cholesky factorization gives a lower triangular matrix $V$, such that $M_k(y^*) = VV^T$. Reduce $V$ to column echelon form

$$U = \begin{bmatrix} 1 & & & & \\ * & & & & \\ 0 & 1 & & & \\ 0 & 0 & 1 & & \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & \cdots & 1 \\ * & * & * & \cdots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & * & \cdots & * \end{bmatrix}$$

by elementary column operations. Notice that the rows of $U$ are indexed by monomials $x^\alpha$ up to degree $k$. Let $\beta_1, \cdots, \beta_r$ be the indices corresponding to the ones in the above $U$. Let $w = [x^{\beta_1}, \cdots, x^{\beta_r}]^T$. Then $[x]_k = Uw$ for all solutions $x = x^*(j)$, $j \in [r]$. Thus for each variable $x_i,\ i \in [n]$, we can extract the $r \times r$ submatrix $N_i$ from $U$ such that

$$N_i w = x_i w, \quad i \in [n].$$

This means $x_i$ is an eigenvalue of $N_i$. Now let $N = \sum_{i=1}^{n} \rho_i N_i$, where $\rho_i \in (0, 1)$ are

random numbers such that $\sum_{i=1}^{n} \rho_i = 1$. Then compute the ordered Schur decomposition $N = QDQ^T$ where $Q = [q_1, \cdots, q_r]$ is orthogonal and $D$ is real and upper triangular with diagonal entries sorted increasingly. Then the extracted solutions are

$$x_i^*(j) = q_j^T N_i q_j, \ i \in [n], j \in [r].$$

### 2.1.3 Example

In this subsection, we present one example to clearly show how to implement Lasserre's SDP relaxation on polynomial optimization problems with only inequality constraints.

Consider the following polynomial optimization problem

$$\begin{cases} f_{\min}^{\text{con}} := \min\limits_{x \in \mathbb{R}^n} & f(x) \\ \text{s.t.} & g_1(x) \geq 0, \ldots, g_\ell(x) \geq 0, \end{cases} \tag{2.7}$$

where $f(x)$ and $g_1(x), \ldots, g_\ell(x)$ are all polynomials in $x$ and their degrees are no greater than $2d$. When $\ell = 0$, problem (2.7) is the standard unconstrained polynomial optimization problem. The $d$-th Lasserre's SDP relaxation ($d$ is also called the relaxation order) for (2.7) is

$$\begin{cases} f_{\text{sos}}^{\text{con}} := \max & \gamma \\ \text{s.t.} & f(x) - \gamma = \sigma_0(x) + g_1(x)\sigma_1(x) + \cdots + g_\ell(x)\sigma_\ell(x), \\ & \sigma_0(x), \sigma_1(x), \ldots, \sigma_\ell(x) \text{ are SOS}, \\ & \deg(\sigma_0), \deg(\sigma_1 g_1), \ldots, \deg(\sigma_\ell g_\ell) \leq 2d. \end{cases} \tag{2.8}$$

Let $N(k) = \binom{n+k}{k}$, $d_i = \lceil \deg(g_i)/2 \rceil$ and $g_0(x) = 1$. Then, $\gamma$ is feasible for (2.8) if and only if there exists $X^{(i)} \in \mathcal{S}^{N(d-d_i)}$ $(i = 0, 1, \ldots, \ell)$ such that

$$f(x) - \gamma = \sum_{i=0}^{\ell} g_i(x)[x]_{d-d_i}^T X^{(i)} [x]_{d-d_i} = \sum_{i=0}^{\ell} X^{(i)} \bullet (g_i(x)[x]_{d-d_i}[x]_{d-d_i}^T),$$
$$X^{(0)} \succeq 0, X^{(1)} \succeq 0, \ldots, X^{(\ell)} \succeq 0.$$

Define constant symmetric matrices $A_\alpha^{(0)}, A_\alpha^{(1)}, \ldots, A_\alpha^{(\ell)}$ such that

$$g_i(x)[x]_{d-d_i}[x]_{d-d_i}^T = \sum_{\alpha \in \mathbb{N}^n : |\alpha| \leq 2d} A_\alpha^{(i)} x^\alpha, \quad i = 0, 1, \ldots, \ell. \tag{2.9}$$

Denote $A_\alpha = (A_\alpha^{(0)}, A_\alpha^{(1)}, \ldots, A_\alpha^{(\ell)})$, $X = (X^{(0)}, X^{(1)}, \ldots, X^{(\ell)})$, and define a cone of products

$$\mathcal{K} = \mathcal{S}_+^{N(d-d_0)} \times \mathcal{S}_+^{N(d-d_1)} \times \cdots \times \mathcal{S}_+^{N(d-d_\ell)}.$$

Recall that $\mathbb{U}_{2d}^n = \{\alpha \in \mathbb{N}^n : 0 < |\alpha| \le 2d\}$. If $f(x) = f_0 + \sum_{\alpha \in \mathbb{U}_{2d}^n} f_\alpha x^\alpha$, then $\gamma$ is feasible for (2.8) if and only if there exists $X$ satisfying

$$\left.\begin{array}{rcll} A_0 \bullet X + \gamma &=& f_0, & \\ A_\alpha \bullet X &=& f_\alpha & \forall\, \alpha \in \mathbb{U}_{2d}^n, \\ X &\in& \mathcal{K}. & \end{array}\right\}$$

Now define $\mathcal{A}, b, C$ as

$$\mathcal{A}(X) = (A_\alpha \bullet X)_{\alpha \in \mathbb{U}_{2d}^n}, \quad b = (f_\alpha)_{\alpha \in \mathbb{U}_{2d}^n}, \quad C = A_0. \tag{2.10}$$

The vector $b$ has dimension $m = N(2d) - 1$. Then, (2.8) is equivalent to the SDP problem up to a constant

$$\left\{\begin{array}{rll} f_{\text{sdp}}^{\text{con}} := \min & C \bullet X & \\ \text{s.t.} & \mathcal{A}(X) = b,\ X \in \mathcal{K}. & \end{array}\right. \tag{2.11}$$

Its dual optimization is

$$\left\{\begin{array}{ll} \max & b^T y \\ \text{s.t.} & \mathcal{A}^*(y) + Z = C,\ Z \in \mathcal{K}. \end{array}\right. \tag{2.12}$$

The adjoint $\mathcal{A}^*(y)$ is defined as

$$\mathcal{A}^*(y) = \sum_{\alpha \in \mathbb{U}_{2d}^n} y_\alpha \operatorname{diag}\left(A_\alpha^{(0)}, A_\alpha^{(1)}, \ldots, A_\alpha^{(\ell)}\right).$$

Note the relation $f_{\text{sos}}^{\text{con}} = -f_{\text{sdp}}^{\text{con}} + f_0 \le f_{\min}^{\text{con}}$.

Suppose $(X^*, y^*, Z^*)$ is an optimal triple for (2.11)-(2.12). Then $-f_{\text{sdp}}^{\text{con}} + f_0$ is a lower bound of the minimum $f_{\min}^{\text{con}}$. The information for minimizers could be obtained from $Z^*$. Note $Z^* = (Z_0^*, Z_1^*, \ldots, Z_\ell^*)$. Since $Z^* \in \mathcal{K}$, every $Z_i^* \succeq 0$. If $Z_0^*$ satisfies FTC (2.6), one or several global minimizers could be obtained by extraction algorithm presented in subsection 2.1.2.

## 2.2 Jacobian SDP Relaxation

In Section 2.1, we have reviewed the standard Lasserre's SDP relaxation method. However, the convergence of Lasserre's SDP relaxations (2.2) and (2.3) might be *asymptotic* for some instances, i.e., only lower bounds are found for each order $k$. To overcome this hurdle, Nie [51] proposed a refined reformulation of (2.1) by a "Jacobian-type" technique whose SDP relaxation has finite convergence. In this section, we first introduce the exact Jacobian SDP relaxation proposed in [51] to solve problem (2.1), then we give a weakened assumption, under which the SDP relaxation in [51] still exact. Some specific examples are present at last.

### 2.2.1 Algorithm Description

Roughly speaking, Jacobian SDP relaxation is to add auxiliary constraints to (2.1) which represent optimality conditions under the assumption that the optimum $f_{\min}$ is achievable. The basic idea is that at each optimizer, the Jacobian matrix of the objective function, the equality constraints and the active inequality constraints must be singular, i.e., all its maximal minors vanish.

Let $m = \min\{m_1 + m_2, n-1\}$. For convenience, denote

$$h := (h_1, \ldots, h_{m_1}) \quad \text{and} \quad g := (g_1, \ldots, g_{m_2}).$$

For a subset $J = \{j_1, \ldots, j_k\} \subseteq [m_2]$, denote $g_J := (g_{j_1}, \ldots, g_{j_k})$. Symbols $\nabla h$ and $\nabla g_J$ represent the gradient vectors of the polynomials in $h$ and $g_J$, respectively. Denote the determinantal variety of $(f, h, g_J)$'s Jacobian being singular by

$$G_J = \left\{ x \in \mathbb{C}^n \mid \text{rank } B^J(x) \leq m_1 + |J| \right\},$$

where

$$B^J(x) = \left[ \nabla f(x) \quad \nabla h(x) \quad \nabla g_J(x) \right].$$

Instead of using all maximal minors to define $G_J$, [51, Section 2.1] discusses how to get the smallest number of defining equations. Let $\eta_1^J, \ldots, \eta_{len(J)}^J$ be the set of defining polynomials for $G_J$ where $len(J)$ is the number of these polynomials. For

each $i = 1, \ldots, len(J)$, define

$$\varphi_i^J(x) = \eta_i^J \cdot \prod_{j \in J^c} g_j(x), \text{ where } J^c = [m_2] \backslash J. \tag{2.13}$$

For simplicity, we list all possible $\varphi_i^J$ in (2.13) sequentially as

$$\varphi_1, \varphi_2, \ldots, \varphi_r, \text{ where } r = \sum_{J \subseteq [m_2], |J| \leq m - m_1} len(J).$$

Define the variety

$$W := \{x \in \mathbb{C}^n \mid \ h_1(x) = \cdots = h_{m_1}(x) = \varphi_1(x) = \cdots = \varphi_r(x) = 0\}. \tag{2.14}$$

We consider the following optimization

$$\begin{cases} f^* := \min\limits_{x \in \mathbb{R}^n} f(x) \\ \qquad \text{s.t. } h_i(x) = 0 \ (i \in [m_1]), \ \varphi_j(x) = 0 \ (j \in [r]), \\ \qquad\quad g_\nu(x) \geq 0, \forall \nu \in \{0,1\}^{m_2}, \end{cases} \tag{2.15}$$

where $g_\nu = g_1^{\nu_1} \cdots g_{m_2}^{\nu_{m_2}}$.

We now construct $N$-th order SDP relaxation [39] for (2.15) and its dual problem. Let $\psi(x)$ be a polynomial with $\deg(\psi) \leq 2N$ and define symmetric matrices $A_\alpha^{(N)}$ such that

$$\psi(x)[x]_d[x]_d^T = \sum_{\alpha \in \mathbb{N}^n : |\alpha| \leq 2N} A_\alpha^{(N)} x^\alpha, \text{ where } d = N - \lceil \deg(\psi)/2 \rceil.$$

Then the $N$-th order localizing moment matrix of $\psi$ is defined as

$$L_\psi^{(N)}(y) = \sum_{\alpha \in \mathbb{N}^n : |\alpha| \leq 2N} A_\alpha^{(N)} y_\alpha, \tag{2.16}$$

where $y$ is a moment vector indexed by $\alpha \in \mathbb{N}^n$ with $|\alpha| \leq 2N$. Denote

$$L_f(y) = \sum_{\alpha \in \mathbb{N}^n : |\alpha| \leq \deg(f)} f_\alpha y_\alpha \quad \text{for} \quad f(x) = \sum_{\alpha \in \mathbb{N}^n : |\alpha| \leq \deg(f)} f_\alpha x^\alpha.$$

The $N$-th order SDP relaxation [39] for (2.15) is the SDP

$$\begin{cases} f_N^{(1)} := \min \ L_f(y) \\ \qquad \text{s.t. } L_{h_i}^{(N)}(y) = 0 \ (i \in [m_1]), \ L_{\varphi_j}^{(N)}(y) = 0 \ (j \in [r]), \\ \qquad\quad L_{g_\nu}^{(N)} \succeq 0, \forall \nu \in \{0,1\}^{m_2}, y_0 = 1. \end{cases} \tag{2.17}$$

Now we present the dual of (2.17). Define the truncated preordering $P^{(N)}$ generated by $g_j$ as

$$P^{(N)} = \left\{ \sum_{\nu \in \{0,1\}^{m_2}} \sigma_\nu g_\nu \,\middle|\, \begin{array}{l} \deg(\sigma_\nu g_\nu) \le 2N \\ \sigma_\nu\text{'s are SOS} \end{array} \right\},$$

and the truncated ideal $I^{(N)}$ generated by $h_i$ and $\varphi_j$ as

$$I^{(N)} = \left\{ \sum_{i=1}^{m_1} \psi_i h_i + \sum_{j=1}^{r} \phi_j \varphi_j \,\middle|\, \begin{array}{l} \deg(\psi_i h_i) \le 2N \ \forall i \\ \deg(\phi_j \varphi_j) \le 2N \ \forall j \end{array} \right\}.$$

It is shown [39] that the dual of (2.17) is the following SDP relaxation for (2.15):

$$\begin{cases} f_N^{(2)} := \max \gamma \\ \qquad \text{s.t. } f(x) - \gamma \in I^{(N)} + P^{(N)}. \end{cases} \tag{2.18}$$

By weak duality, we have $f_N^{(2)} \le f_N^{(1)} \le f^*$. For any subset $J = \{j_1, \ldots, j_k\} \subseteq [m_2]$, let

$$V(h, g_J) = \{x \in \mathbb{C}^n \mid h_i(x) = 0,\ g_j(x) = 0,\ i = 1, \ldots, m_1,\ j \in J\}.$$

We make the following assumption.

**Assumption 2.2.1.** (i) $m_1 \le n$. (ii) *For any feasible point $u$, at most $n - m_1$ of $g_1(u), \ldots, g_{m_2}(u)$ vanish.* (iii) *For every $J = \{j_1, \ldots, j_k\} \subseteq [m_2]$ with $k \le n - m_1$, the Jacobian $[\nabla h \quad \nabla g_J]$ has full rank on $V(h, g_J)$.*

Under the above assumption, the following main result is shown in [51].

**Theorem 2.2.2.** ([51, Theorem 2.3]) *Suppose Assumption 2.2.1 holds. Then $f^* > -\infty$ and there exists $N^* \in \mathbb{N}$ such that $f_N^{(1)} = f_N^{(2)} = f^*$ for all $N \ge N^*$. Furthermore, if the minimum $f_{min}$ of (2.1) is achievable, then $f_N^{(1)} = f_N^{(2)} = f_{min}$ for all $N \ge N^*$.*

**Algorithm 2.2.3.** (Jacobian SDP relaxation)

***Input:*** *Objective function $f(x)$, constraints functions $h_i(x), g_j(x)$, maximal relaxation order $k_{\max}$.*

***Output:*** *Global minimum and minimizers of problem (2.1).*

I *Construct the auxiliary polynomials $\varphi_l(x)$'s.*

II *Set* $d := \max\{1, d_f, d_{h_i}, d_{g_j}, d_{\varphi_l}\}$ *and initial relaxation order* $k = d$.

III *Solve (2.15) by Algorithm 2.1.3.*

IV *For* $t \in [d, k]$, *check condition (2.6).*

1 *If (2.6) holds for some* $t$, *get minimizers by Extraction Algorithm [18] and stop;*

2 *Otherwise, go to Step V.*

V *If* $k > k_{\max}$, *stop; otherwise, set* $k = k + 1$ *and go to Step III.*

In contrast to Lasserre's SDP relaxation, Jacobian SDP relaxation is more complicated due to the auxiliary polynomials $\varphi_l(x)$'s. We refer to [51, Section 4] for some simplified versions of Jacobian SDP relaxation method.

### 2.2.2 Weakened Convergence Condition

According to Theorem 2.2.2, it is possible to solve the polynomial optimization (2.1) exactly by a single SDP relaxation, and it is also shown in [51] that Assumption 2.2.1 is generically true. In this subsection, we prove that the condition (iii) in Assumption 2.2.1 can always be weakened such that the conclusions in Theorem 2.2.2 still holds, i.e., the Jacobian SDP relaxation [51] is still exact under the weakened Assumption 2.2.5.

**Definition 2.2.4.** *For every set* $J = \{j_1, \ldots, j_k\} \subseteq [m_2]$ *with* $k \leq n - m_1$, *let*

$$\Theta_J = \{x \in V(h, g_J) \mid \textit{rank } [\nabla h \quad \nabla g_J] < m_1 + |J|\} \quad \textit{and} \quad \Theta = \bigcup_{J \subseteq [m_2], |J| \leq n - m_1} \Theta_J.$$

**Assumption 2.2.5.** (i) $m_1 \leq n$. (ii) *For any* $u \in S$, *at most* $n - m_1$ *of* $g_1(u), \ldots, g_{m_2}(u)$ *vanish.* (iii) *The set* $\Theta$ *is finite.*

Let $K$ be the variety defined by the KKT conditions

$$K = \left\{ (x, \lambda, \mu) \in \mathbb{C}^{n+m_1+m_2} \;\middle|\; \begin{array}{l} \nabla f(x) = \sum_{i=1}^{m_1} \lambda_i \nabla h_i(x) + \sum_{j=1}^{m_2} \mu_j \nabla g_j(x) \\ h_i(x) = \mu_j g_j(x) = 0, \forall (i,j) \in [m_1] \times [m_2] \end{array} \right\}$$

and

$$K_x = \{x \in \mathbb{C}^n \mid (x, \lambda, \mu) \in K \text{ for some } \lambda, \mu\}.$$

Under Assumption 2.2.1, [51, Lemma 3.1] states that $W = K_x$. We now improve this result as follows.

**Lemma 2.2.6** (Revised Version of Lemma 3.1 in [51])**.** *Under conditions* $(i)$ *and* $(ii)$ *in Assumption 2.2.5,* $W = \Theta \cup K_x$.

*Proof.* The proof of [51, Lemma 3.1] shows that $W \backslash \Theta \subseteq K_x \subseteq W$. With a similar argument, we prove $\Theta \subseteq W$. Recall that $B^J = [\nabla f(x) \quad \nabla h(x) \quad \nabla g_J(x)]$. Choose an arbitrary $u \in \Theta$ and let $u \in \Theta_I$ for some $I \subseteq [m_2]$. If $I = \emptyset$, then $[\nabla h]$ and $B^J(u)$ are both singular for any $J \subseteq [m_2]$, which implies $\varphi_i(u) = 0$ and $u \in W$. If $I \neq \emptyset$, write $I = \{i_1, \ldots, i_t\}$. Let $J = \{j_1, \ldots, j_k\} \subseteq [m_2]$ be an arbitrary index set with $m_1 + k \leq m$.

**Case** $I \nsubseteq J$ At least one $j \in J^c$ belongs to $I$. By the choice of $I$ and the definition of $\varphi_i(x)$,

$$\varphi_i^J(u) = \eta_i^J \cdot \prod_{j \in J^c} g_j(u) = 0.$$

**Case** $I \subseteq J$ Then $[\nabla h \quad \nabla g_I]$ and $[\nabla f(x) \quad \nabla h(x) \quad \nabla g_J(x)]$ are both singular. Hence, all polynomials $\varphi_i^J(x)$'s vanish at $u$.

Combining the above two cases, we have all $\varphi_i^J(x)$ vanish at $u$. Thus, $u \in W$ which implies $W = \Theta \cup K_x$. □

**Lemma 2.2.7.** *Under conditions* (i) *and* (ii) *in Assumption 2.2.5, if the minimum* $f_{min}$ *of* (2.1) *is achievable, then* $f^* = f_{min}$.

*Proof.* By the construction of (2.15), $f^* \geq f_{\min}$. Suppose $f_{\min} = f(x^*)$ where $x^*$ is a feasible point of (2.1). If $x^* \notin \Theta$, then the linear independence constraint qualification (LICQ) is satisfied at $x^*$ which implies $x^* \in K_x$ [56, Theorem 12.1]. Since $W = \Theta \cup K_x$ by Lemma 2.2.6, we have $x^* \in W$ which implies $f^* = f_{\min}$. □

Next we show that the conclusion in [51, Lemma 3.2] still holds under Assumption 2.2.5.

**Lemma 2.2.8** (Revised Version of Lemma 3.2 in [51]). *Suppose Assumption 2.2.5 holds. Let* $T = \{x \in \mathbb{R}^n \mid g_j(x) \geq 0,\ j = 1, \ldots, m_2\}$. *Then there exist disjoint subvarieties* $W_0, W_1, \ldots, W_r$ *of* $W$ *and distinct* $v_1, \ldots, v_r \in \mathbb{R}$ *such that*

$$W = W_0 \cup W_1 \cup \cdots \cup W_r, \quad W_0 \cap T = \emptyset, \quad W_i \cap T \neq \emptyset, \quad i = 1, \ldots, r,$$

*and* $f(x)$ *is constantly equal to* $v_i$ *on* $W_i$ *for* $i = 1, \ldots, r$.

*Proof.* Denote $Zar(K_x)$ the Zariski closure of $K_x$ and let $\Omega = W \backslash Zar(K_x)$. By Lemma 2.2.6, we have $Zar(K_x) \subseteq W$ and $\Omega \subseteq \Theta$. With the proof of [51, Lemma 3.2], we can conclude that there exist disjoint subvarieties $W_0, W_1, \ldots, W_t$ of $Zar(K_x)$ and distinct $v_1, \ldots, v_t \in \mathbb{R}$ such that

$$Zar(K_x) = W_0 \cup W_1 \cup \cdots \cup W_t, \quad W_0 \cap T = \emptyset, \quad W_i \cap T \neq \emptyset, \quad i = 1, \ldots, t,$$

and $f(x)$ is constantly equal to $v_i$ on $W_i$ for $i = 1, \ldots, t$. We now consider the set $\Omega$. Let $W_0 = V(E_0)$, then for any $u \in \Omega \cap \mathbb{C}^n$, $W_0 \cup \{u\} = V(E_0) \cup V(\langle x - u \rangle) = V(\langle x - u \rangle \cdot E_0)$. Since $\Omega \cap \mathbb{C}^n \subseteq \Theta$ is a finite set by Assumption 2.2.5, if we group $W_0$ and $\Omega \cap \mathbb{C}^n$ together, then we get a new subvariety. We still denote it by $W_0$ for convenience. Then $W_0 \cap T = \emptyset$. Take any $w \in \Omega \cap \mathbb{R}^n$, if $f(w) = v_{i_0}$ for some $i_0 \in \{1, \ldots, t\}$, then we put $w$ into $W_{i_0}$ and get a new subvariety by the same reason as $W_0$. We still write the resulting subvariety as $W_{i_0}$. If for any $i \in \{1, \ldots, t\}$, $f(w) \neq v_i$, then let $W_{t+1} = \{w\}$ and $v_{t+1} = f(w) \in \mathbb{R}$. Since $\Omega \cap \mathbb{R}^n \subseteq \Theta$ is a finite set, the above process will terminate and we can obtain the required decomposition of $W$. □

Since we get the same result as in [51, Lemma 3.2] under the weakened Assumption 2.2.5, [51, Theorem 3.4] which is based on [51, Lemma 3.2] can be restated as follows.

**Theorem 2.2.9** (Revised Version of Theorem 3.4 in [51]). *Suppose Assumption 2.2.5 holds. Then* $f^* > -\infty$ *and there exists* $N^* \in \mathbb{N}$ *such that for all* $\varepsilon > 0$

$$f(x) - f^* + \varepsilon \in I^{(N^*)} + P^{(N^*)}. \tag{2.19}$$

Since $\varepsilon$ in (2.19) is arbitrary, by Lemma 2.2.7, Theorem 2.2.2 becomes

**Theorem 2.2.10** (Revised Version of Theorem 2.3 in [51]). *Suppose Assumption 2.2.5 holds. Then $f^* > -\infty$ and there exists $N^* \in \mathbb{N}$ such that $f_N^{(1)} = f_N^{(2)} = f^*$ for all $N \geq N^*$. Furthermore, if the minimum $f_{min}$ of* (2.1) *is achievable, then $f_N^{(1)} = f_N^{(2)} = f_{min}$ for all $N \geq N^*$.*

**Remark 2.2.11.** We now compare the conditions (iii) in Assumption 2.2.1 and 2.2.5. For any $J = \{j_1, \ldots, j_k\} \subseteq [m_2]$ with $k \leq n - m_1$, suppose the ideal $\langle h, g_J \rangle$ is radical and its codimension is $m_1 + |J|$. Then the condition (iii) in Assumption 2.2.1 requires the variety $V(h, g_J)$ is nonsingular for every subset $J$. In this section, we have proved that if the singularities of $V(h, g_J)$ are finite, i.e. the condition (iii) in Assumption 2.2.5 holds, the Jacobian SDP relaxation [51] is still exact.

**Corollary 2.2.12.** *Suppose that*

(a) *For each subset $J \subseteq [m_2]$ with $|J| \leq n - m_1$, $\langle h, g_J \rangle$ is a radical ideal and its codimension is $m_1 + |J|$;*

(b) *$V(h)$ is a smooth variety of dimension $\leq 2$.*

*Then the condition* (iii) *in Assumption 2.2.5 always holds. Therefore, if conditions* (i) *and* (ii) *in Assumption 2.2.5 are satisfied, then the conclusions of Theorem 2.2.10 hold.*

*Proof.* For any subset $J \subseteq [m_2]$ with $|J| \leq n - m_1$, by (a), $\Theta_J$ is the set of singularities of $V(h, g_J)$. If $J = \emptyset$, then $\Theta_J = \emptyset$ by (b). If $J \neq \emptyset$, then by [28, Proposition 3.3.14], $\dim \Theta_J < \dim V(h, g_J)$. Since $\dim V(h, g_J) \leq 1$ by (a) and (b), $\Theta_J$ is a finite set for each $J \subseteq [m_2]$ with $|J| \leq n - m_1$. Thus the condition (iii) in Assumption 2.2.5 always holds. □

We now give an example to illustrate the finite convergence of the Jacobian SDP relaxation [51] under the weakened Assumption 2.2.5.

**Example 2.2.13.** Consider the following polynomial optimization

$$\begin{cases} \min\limits_{x_1, x_2 \in \mathbb{R}} f(x_1, x_2) := x_1 x_2^2 + x_1 \\ \quad \text{s.t. } h(x_1, x_2) := -x_1^3 + x_2^2 = 0. \end{cases}$$

Clearly, the minimum $f_{\min} = 0$ is achieved at $(0, 0)$. However, it is easy to verify that $(0, 0)$ is a singular point and does not satisfy the KKT conditions. Since $(0, 0)$ is the only real singularity, Assumption 2.2.5 holds which is also guaranteed by Corollary 2.2.12. In the following, we show the finite convergence of the Jacobian SDP relaxation [51] by giving the exact equation (2.19).

By the construction of (2.15), $m_1 = 1, m_2 = 0$ and $r = 1$. $\varphi(x_1, x_2) := 2x_2(x_2^2 + 1) + 6x_1^3x_2$. For any $\varepsilon > 0$, let

$$\sigma_0(x_1, x_2) := 16\left(\varepsilon + \frac{(x_1x_2^2 + x_1 + 1)^2}{4} + (x_1x_2^2 + x_1 - 1)^2x_2^2\right)x_1^6 + (4x_1^3 + 1)^2\varepsilon \left(1 + \frac{x_1x_2^2 + x_1}{2\varepsilon} - \frac{(x_1x_2^2 + x_1)^2}{8\varepsilon^2}\right)^2.$$

$$\begin{aligned}
\psi(x_1, x_2) := {} & 8x_1 + 8\varepsilon - 12x_1^8x_2^4 - 24x_1^8x_2^2 + 24x_1^7x_2^2 + 8x_1x_2^2 + 4x_1^3 + 32\varepsilon x_1^3 - \frac{x_1^3}{\varepsilon^2} \\
& + \frac{x_1^4}{8\varepsilon^3} - \frac{2x_1^6}{\varepsilon^2} + \frac{x_1^7}{4\varepsilon^3} + 8\varepsilon x_2^2 + \frac{x_1x_2^2}{64\varepsilon^3} - \frac{x_2^2}{8\varepsilon^2} + \frac{x_1}{64\varepsilon^3} - \frac{1}{8\varepsilon^2} + 4x_1^3x_2^2 \\
& + 4x_1^5x_2^2 + 4x_1^5 + 24x_1^4 - \frac{243x_1^{10}x_2^2}{256\varepsilon^3} - \frac{3x_1^{10}x_2^6}{16\varepsilon^3} - \frac{45x_1^7x_2^4}{128\varepsilon^3} - \frac{311x_1^7x_2^2}{1024\varepsilon^3} \\
& - \frac{3x_1^7x_2^6}{32\varepsilon^3} + \frac{3x_1^3x_2^4}{32\varepsilon^2} + \frac{33x_1^9x_2^2}{8\varepsilon^2} + \frac{29x_1^6x_2^2}{32\varepsilon^2} - \frac{45x_1^4x_2^4}{1024\varepsilon^3} - \frac{45x_1^{10}x_2^4}{64\varepsilon^3} - \frac{17x_1^3x_2^2}{32\varepsilon^2} \\
& + \frac{3x_1^9x_2^4}{2\varepsilon^2} + \frac{3x_1^6x_2^4}{4\varepsilon^2} + \frac{47x_1^4x_2^2}{1024\varepsilon^3} - \frac{3x_1^4x_2^6}{256\varepsilon^3}.
\end{aligned}$$

$$\begin{aligned}
\phi(x_1, x_2) := {} & - \frac{x_1^{10}x_2^5}{32\varepsilon^3} - \frac{15x_1^{10}x_2^3}{128\varepsilon^3} + \frac{x_1^9x_2^3}{4\varepsilon^2} - \frac{x_1^7x_2^5}{64\varepsilon^3} - \frac{81x_1^{10}x_2}{512\varepsilon^3} - 2x_1^8x_2^3 + \frac{11x_1^9x_2}{16\varepsilon^2} \\
& - \frac{15x_1^7x_2^3}{256\varepsilon^3} - 4x_1^8x_2 + \frac{x_1^6x_2^3}{8\varepsilon^2} - \frac{x_1^4x_2^5}{512\varepsilon^3} - \frac{337x_1^7x_2}{2048\varepsilon^3} + 4x_1^7x_2 + \frac{59x_1^6x_2}{64\varepsilon^2} \\
& - \frac{15x_1^4x_2^3}{2048\varepsilon^3} - 2x_1^5x_2 + \frac{x_1^3x_2^3}{64\varepsilon^2} - \frac{x_1^4x_2}{16\varepsilon^3} + \frac{7x_1^3x_2}{16\varepsilon^2} - 2x_1^3x_2 - 4x_1x_2 - \frac{x_1x_2}{128\varepsilon^3} \\
& + \frac{x_2}{16\varepsilon^2} - 4\varepsilon x_2.
\end{aligned}$$

It can be verified that

$$f(x_1, x_2) + \varepsilon = \sigma_0(x_1, x_2) + \psi(x_1, x_2)h(x_1, x_2) + \phi(x_1, x_2)\varphi(x_1, x_2).$$

Since each term on the right side of the above equation has degree $\leq 20$, we take $N^* = 10$ in (2.19). Because $\sigma_0(x_1, x_2)$ is a sum of squares of polynomials, we have $\sigma_0(x_1, x_2) \in P^{(10)}$ and $\psi(x_1, x_2)h(x_1, x_2) + \phi(x_1, x_2)\varphi(x_1, x_2) \in I^{(10)}$. Therefore, $f(x_1, x_2) + \varepsilon \in I^{(10)} + P^{(10)}$ for any $\varepsilon > 0$. Hence, we have $f_N^{(1)} = f_N^{(2)} = f_{\min} = 0$ for all $N \geq 10$.

As an application, [53, Corollary 4.2] also points out that if (2.1) has a nonempty set of finitely many global minimizers and Assumption 2.2.1 is satisfied, then the flat truncation is always satisfied for the hierarchy of Jacobian SDP relaxations. Since we have proved that Assumption 2.2.1 can be weakened as Assumption 2.2.5, we have

**Corollary 2.2.14** (Revised Version of Corollary 4.2 in [53])**.** *Suppose* (2.1) *has a nonempty set of finitely many global minimizers and Assumption 2.2.5 is satisfied. Then, for all $N$ big enough, the optimal value of* (2.18) *equals the global minimum of* (2.1) *and every minimizer of* (2.17) *has a flat truncation.*

### 2.2.3 Examples

To clearly see how to define the redundant equations, in the following, we give some examples with specific constraints.

**Example 2.2.15.** (Feasible set $K = \mathbb{R}^n$)

Consider the following polynomial optimization problem:

$$\min_{x\in\mathbb{R}^n} \quad f(x). \tag{2.20}$$

There is no constraint. The KKT condition is just $\nabla_x f(x) = 0$, i.e., there are $n$ redundant polynomials defined as:

$$\varphi_1(x) = \tfrac{\partial f(x)}{\partial x_1}, \ldots, \varphi_n(x) = \tfrac{\partial f(x)}{\partial x_n}. \tag{2.21}$$

**Example 2.2.16.** (Feasible set $K$ is a ball)

Consider the following polynomial optimization problem:

$$\begin{cases} \min\limits_{x\in\mathbb{R}^n} f(x) \\ \text{s.t. } \|x\|^2 \le R^2. \end{cases} \tag{2.22}$$

There is only one inequality constraint. By using Jacobian method, there are $3n-3$ redundant polynomials defined as follows:

$$\begin{aligned} &(R^2 - \|x\|^2)\, \tfrac{\partial f(x)}{\partial x_i} = 0, \ i = 1, \ldots, n, \\ &\sum_{i+j=\ell} \Big( \tfrac{\partial f(x)}{\partial x_i} x_j - x_i \tfrac{\partial f(x)}{\partial x_j} \Big) = 0, \quad \ell = 3, \ldots, 2n-1. \end{aligned} \tag{2.23}$$

**Example 2.2.17.** (Feasible set $K$ is a box)

Consider the following polynomial optimization problem:

$$\begin{cases} \min\limits_{x\in\mathbb{R}^n} f(x) \\ \text{s.t. } a \le x \le b, \end{cases} \tag{2.24}$$

where $a, b \in \mathbb{R}^n$.

There are $n$ interval constraints, and each interval is equivalent to an inequality constraint $\psi_i(x) = (a_i + b_i)x_i - x_i^2 - a_i b_i \ge 0,\ i \in [n]$. The number of active constraint $|J|$ has $n$ possibilities. When $|J| = 0$, there are $n$ redundant polynomials: $\varphi_j(x) = \left(\prod\limits_{i=1}^{n} \psi_i(x)\right) \frac{\partial f(x)}{\partial x_j} = 0,\ j \in [n]$. When $|J| = 1$, the $n$ constraints have the same possibility to be active constraint. If $\psi_j(x)$ is the chosen active constraint, there are $2n - 1 - j$ redundant polynomials: $\varphi_\ell(x) = \left(\prod\limits_{i=1, i\neq j}^{n} \psi_i(x)\right) \frac{\partial f(x)}{\partial x_{\ell+2-j}} (a_j + b_j - 2x_j) = 0$ for $\ell = j-1, \ldots, 2n-3$. List all possible $|J|$ in sequence, and we get all redundant polynomials.

**Example 2.2.18.** (Feasible set $K$ is a simplex)

Consider the following polynomial optimization problem:

$$\begin{cases} \min\limits_{x\in\mathbb{R}^n} f(x) \\ \text{s.t. } \sum\limits_{i=1}^{n} x_i = 1, \quad x_i \ge 0, \quad i = 1, \ldots, n. \end{cases} \tag{2.25}$$

There is one equality constraint in problem (2.25), so the number of active constraints $1 \le |J| \le n-1$. Denote the first equality constraint as $\psi_0(x) = \sum\limits_{i=1}^{n} x_i - 1 = 0$, and the other $n$ constraints as $\psi_i(x) = x_i \ge 0,\ i \in [n]$. When $|J| = 1$, there is no inequality constraint to be active, there are $2n - 3$ redundant polynomials $\varphi_\ell(x) = \left(\prod\limits_{i=1}^{n} x_i\right) \cdot \sum\limits_{i+j=\ell+2} \left(\frac{\partial f(x)}{\partial x_i} - \frac{\partial f(x)}{\partial x_j}\right) = 0,\ \ell = 1, \ldots, 2n-3$. Similarly, when $|J| > 1$, there are $|J| - 1$ inequalities to be active, choose $|J| - 1$ active inequalities, denote as $A$, and redenote the rest $[n] \setminus A$ inactive variables as $x_1, \ldots, x_{n+1-|J|}$, there are $2(n - |J|) - 1$ redundant polynomials, which are defined as $\varphi_\ell(x) = \left(\prod\limits_{k=1}^{n+1-|J|} x_i\right) \cdot \sum\limits_{i+j=\ell+2} \left(\frac{\partial f(x)}{\partial x_i} - \frac{\partial f(x)}{\partial x_j}\right) = 0,\ \ell = 1, \ldots, 2n-2|J|-1$. There are totally $r = \sum\limits_{|J|=1}^{n-1} \binom{n}{|J|-1} \cdot (2n - 2|J| - 1)$ redundant polynomials.

## 2.3 Large Scale Polynomial Optimization

In this section, we study how to solve Lasserre's SDP relaxations for large scale polynomial optimization.

### 2.3.1 Interior Point Method vs. Regularization Method

Consider the standard SDP problem (1.4) and its dual problem (1.5). Let $X$ be optimal for (1.4) and $(y, Z)$ be optimal for (1.5), then the triple $(X, y, Z)$ satisfies the optimality condition

$$\left.\begin{aligned} \mathcal{A}(X) &= b \\ \mathcal{A}^*(y) + Z &= C \\ X, Z \succeq 0,\ X \bullet Z &= 0 \end{aligned}\right\}. \tag{2.26}$$

For interior point method, it generates a sequence $\{(X_k, y_k, Z_k)\}$ converging to an optimal triple. At each step, a search direction $(\Delta X, \Delta y, \Delta Z)$ needs to be computed. To compute $\Delta y$, typically an $m \times m$ linear system needs to be solved. To compute $\Delta X$ and $\Delta Z$, two linear matrix equations need to be solved. The cost for computing $\Delta y$ is $\mathcal{O}(m^3)$. When $m = \mathcal{O}(N)$, the cost for computing $\Delta y$ is $\mathcal{O}(N^3)$. In this case, solving SDP is not very expensive if $N$ is not too big (like less than $1,000$). However, when $m = \mathcal{O}(N^2)$, the cost for computing $\Delta y$ would be $\mathcal{O}(N^6)$, which is very expensive even for moderately large $N$ (like 500). In this case, computing $\Delta y$ is very expensive. It requires storing a matrix of dimension $m \times m$ in computer and $\mathcal{O}(m^3)$ arithmetic operations.

Unfortunately, SDP relaxations arising from polynomial optimization belong to the bad case that $m = \mathcal{O}(N^2)$, which is why the SDP solvers based on interior point methods have difficulty in solving big polynomial optimization (like degree 4 with 100 variables). We explain why this is the case by unconstrained polynomial optimization. Let $p(x)$ be a polynomial of degree $2d$. Then, $p(x)$ is SOS if and only if there exists $X \succeq 0$ [61] such that $p(x) = [x]_d^T X [x]_d$, where $[x]_d$ denotes the column vector of all monomials up to degree $d$ in graded lexicographical ordering. Note the length of $[x]_d$ is $N = \binom{n+d}{d}$. If we write

**Table 2.1**: A list of size of SDP (2.27).

(In each pair $(N, m)$, $N$ is the length of matrix and $m$ is the number of equality constraints.)

| | | | | |
|---|---|---|---|---|
| n= 2d = 4 | 20 (231,10625) | 30 (496,46375) | 40 (861,135750) | 50 (1326,316250) |
| n= 2d=4 | 60 (1891,635375) | 70 (2556,1150625) | 80 (3321,1929500) | 90 (4186,3049500) |
| n= 2d = 6 | 15 (816,54263) | 20 (1771,230229) | 25 (3276,736280) | 30 (5456,1947791) |
| n= 2d = 8 | 10 (1001,43757) | 15 (3876,490313) | 20 (10626,3108104) | 25 (23751,13884155) |
| n= 2d = 10 | 8 (1287,43757) | 9 (2002,92377) | 10 ( 3003,184755) | 15 (15504,3268759) |

$p(x) = \sum_{\alpha\in\mathbb{N}^n:|\alpha|\le 2d} p_\alpha x_1^{\alpha_1}\cdots x_n^{\alpha_n}$, then $p(x)$ being SOS is equivalent to the existence of a symmetric $N \times N$ matrix $X$ satisfying

$$\begin{array}{rcl} A_\alpha \bullet X &=& p_\alpha \quad \forall\, \alpha \in \mathbb{N}^n : |\alpha| \le 2d, \\ X &\succeq& 0. \end{array} \tag{2.27}$$

Here $A_\alpha$ are certain constant symmetric matrices. The number of equalities is $m = \binom{n+2d}{2d}$. For any fixed $d$, $m = \mathcal{O}(n^{2d}) = \mathcal{O}(N^2)$. The size of SDP (2.27) is huge for moderately large $n$ and $d$. Table 2.1 lists the size of SDP (2.27) for some typical values of $(n, 2d)$. In Table 2.1, each pair $(N, m)$, $N$ is the length of matrix and $m$ is the number of equality constraints.

As we have seen earlier, when interior point type methods are applied to solve (1.4)-(1.5), at each step we need to solve an $m \times m$ linear system and two matrix equations. To compute $\Delta y$, we need to store an $m \times m$ matrix and implement $\mathcal{O}(n^{6d})$ arithmetic operations. This is very expensive for even moderately large $n$ and $d$, and hence severely limits the solvability of SDP relaxations in polynomial optimization. For instance, on a regular computer, to solve a general quartic polynomial optimization, it is almost impossible to apply interior point methods when there are more than 20 variables.

For regularization method, in each step, only $\Delta y$ needs to be computed, which is well designed to solve SDP problems whose number of equality constraints

$m$ is significantly bigger than the matrix length $N$. The numerical experiments in [26, 27, 81] show that these methods are practical and efficient in solving large scale SDP problems. In next subsection, we present numerical examples to show its efficiency on solving polynomial optimization problems. By regularization methods, significantly bigger problems could be solved on a regular computer, which is almost impossible by interior point method.

### 2.3.2 Numerical Experiments

This subsection presents some numerical examples of applying regularization method (i.e., Algorithm 1.3.5) on solving polynomial optimization by Lasserre's SDP relaxation with the lowest relaxation order. An excellent implementation of Algorithm 1.3.5 is software `SDPNAL` [83]. We use it to solve the SDP relaxations (its earlier version in 2010 was used). The computation is implemented with Matlab 7.10 on a Dell 64-bit Linux Desktop running CentOS (5.6) with 8GB memory and Intel(R) Core(TM) i7 CPU 860 2.8GHz. We use the following parameters of `SDPNAL` $\sigma_0 = 10,\ K = 500,\ \texttt{Tol} = 10^{-6}$. Set

$$R_P = \frac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2}, \quad R_D = \frac{\|\mathcal{A}^*(y) + Z - C\|_2}{1 + \|C\|_2},$$

which measure the feasibilities of the computed solutions for the primal and dual problems respectively. We terminate the algorithm when $\max\{R_P, R_D\} \leq \texttt{Tol}$. Other parameters are set to be the default ones of `SDPNAL`.

If the computed dual optimal solution $Z_0^*$ of (2.3) or (2.12) satisfies FTC (2.6), we extract one or several global minimizers $x^*$ by using the Extraction Algorithm [18] reviewed in subsection 2.1.2; otherwise, we just set $Z_0^*(2 : n+1, 1)$ as a starting point and get a local optimal solution $x^*$ by using nonlinear programming methods (e.g., use Optimization Toolbox in Matlab). In either case, the error of computed $x^*$ is measured as

$$\text{errsol} \quad = \quad \frac{|f(x^*) - \underline{f}|}{\max\{1, |f(x^*)|\}}, \tag{2.28}$$

where $\underline{f}$ is a lower bound returned by solving the SDP relaxation. The error of a

computed optimal triple $(X, y, Z)$ for SDP relaxation itself is measured as

$$\text{errsdp} = \max\left\{\frac{|b^\top y - \langle C, X\rangle|}{1 + |b^\top y| + |\langle C, X\rangle|}, R_P, R_D\right\}. \tag{2.29}$$

The consumed computer time is in the format `hr:mn:sc` with hr (resp. `mn`, `sc`) stands for the consumed hours (resp. minutes, seconds). In the tables of this thesis, `min`, `med` and `max` respectively stands for the minimum, median, and maximum of quantities like time, solution error, etc.

The testing problems in our experiments are in two categories: (a) random unconstrained polynomial optimization; (b) random constrained polynomial optimization.

**Example 2.3.1** (Random Unconstrained Polynomial Optimization)**.**

We test the performance of Algorithm 1.3.5 (implemented by `SDPNAL` [83]) in solving SDP relaxations for random polynomial optimization. To ensure the existence of a finite global minimum, we generate $f(x)$ randomly as

$$f(x) = f^T[x]_{2d-1} + [x^d]^T F^T F[x^d],$$

where $f/F$ is a Gaussian random vector/matrix of a proper dimension. Here $[x^d]$ denotes the vector of monomials of degree equal to $d$. The computational results are shown in Tables 2.2-2.5. There $(N, m)$ denotes the size of the corresponding SDP relaxation (2.2)-(2.3). If $N < 1000$, we test 20 instances randomly; if $N \in [1000, 1500]$, we test 10 instances randomly; and if $N > 1500$, we test 3 instances randomly.

When $f(x)$ has degree 4 ($d = 2$), SDP relaxation (2.2)-(2.3) is solved quite well. For $n = 20 \sim 30$, the computation takes up to half a minute; for $n = 40 \sim 60$, it takes a couple of minutes; for $n = 70 \sim 80$, it takes less than one hour; for $n = 90 \sim 100$, it takes a few hours. When $f(x)$ has degree 6 ($d = 3$), for $n = 15$, solving (2.2)-(2.3) takes up to a few hours; for $n = 20 \sim 25$, it takes a couple of hours. When $f(x)$ has degree 8 ($d = 4$), for $n = 10$, solving (2.2)-(2.3) takes a couple of minutes; for $n = 12 \sim 15$, it takes about one to ten hours. When $f(x)$ has degree 10 ($d = 5$), for $n = 8$, solving (2.2)-(2.3) takes a couple of minutes; for $n = 9$, it takes less than one hour; for $n = 10$, it takes a few hours. From

**Table 2.2**: Computational results for random Example 2.3.1 of degree 4

| $n$ | (N,m) | time(min,max) | | errsol(min,max) | errsdp(min,max) |
|---|---|---|---|---|---|
| 20 | (231,10625) | 0:00:02 | 0:00:09 | (4.1e-7, 1.6e-4) | (2.5e-7, 1.3e-6) |
| 30 | (496,46375) | 0:00:12 | 0:00:31 | (1.3e-7, 1.5e-4) | (3.2e-7, 1.0e-6) |
| 40 | (861,135750) | 0:00:57 | 0:01:24 | (7.8e-7, 3.1e-4) | (4.2e-7, 9.6e-7) |
| 50 | (1326,316250) | 0:02:44 | 0:04:08 | (1.3e-5, 2.3e-4) | (5.6e-7, 8.3e-7) |
| 60 | (1891,635375) | 0:07:55 | 0:09:48 | (4.6e-5, 5.1e-4) | (4.8e-7, 9.5e-7) |
| 70 | (2556,1150625) | 0:17:38 | 0:22:33 | (8.0e-5, 3.3e-4) | (4.1e-7, 9.2e-7) |
| 80 | (3321,1929500) | 0:38:45 | 0:42:46 | (9.3e-5, 9.6e-4) | (3.7e-7, 9.9e-7) |
| 90 | (4186,3049500) | 1:37:04 | 2:02:01 | (1.1e-4, 6.4e-4) | (4.3e-7, 9.5e-7) |
| 100 | (5151,4598125) | 2:48:03 | 3:35:27 | (2.1e-4, 4.5e-4) | (7.1e-7, 8.7e-7) |

Tables 2.2 to 2.5, we can see that the SDP relaxations are solved successfully. The obtained solutions for polynomial optimization are also reasonably very well. They are slightly less accurate than the computed solutions of the SDP relaxations themselves. This is probably because the SDP relaxation (2.2)-(2.3) is not exact in minimizing the generated polynomials.

**Table 2.3**: Computational results for random Example 2.3.1 of degree 6

| $n$ | (N,m) | time(min,max) | | errsol(min,max) | errsdp(min,max) |
|---|---|---|---|---|---|
| 10 | (286,8007) | 0:00:07 | 0:00:36 | (2.7e-7, 6.6e-5) | (2.4e-8, 1.1e-6) |
| 15 | (816,54263) | 0:01:12 | 3:07:37 | (5.1e-6, 7.0e-5) | (2.0e-7, 9.6e-7) |
| 20 | (1771,230229) | 2:54:42 | 15:10:08 | (1.4e-4, 4.0e-4) | (3.1e-7, 6.0e-7) |
| 25 | (3276,736280) | 2:02:59 | 7:34:03 | (1.6e-3, 4.7e-2) | (2.6e-6, 5.7e-5) |

**Table 2.4**: Computational results for random Example 2.3.1 of degree 8

| $n$ | (N,m) | time(min,max) | | errsol(min,max) | errsdp(min,max) |
|---|---|---|---|---|---|
| 8 | (495,12869) | 0:00:18 | 0:01:11 | (1.6e-7, 5.6e-4) | (1.0e-7, 4.1e-6) |
| 10 | (1001,43757) | 0:04:46 | 0:08:05 | (3.9e-5, 5.3e-4) | (2.4e-7, 3.0e-6) |
| 12 | (1820,125969) | 0:26:32 | 1:02:37 | (1.3e-5, 5.7e-3) | (1.1e-7, 5.3e-6) |
| 15 | (3876,490313) | 6:31:11 | 10:21:21 | (6.8e-4, 4.5e-3) | (9.9e-7, 5.6e-6) |

The computations here show that Algorithm 1.3.5 would solve large scale polynomial optimization. A quartic polynomial optimization with 100 variables would be solved within a couple of hours on a regular computer. This is almost impossible by using SDP solvers based on interior point methods. □

**Table 2.5**: Computational results for random Example 2.3.1 of degree 10

| $n$ | (N,m) | time(min,max) | | errsol(min,max) | errsdp(min,max) |
|---|---|---|---|---|---|
| 6 | (462,8007) | 0:00:10 | 0:00:32 | (3.6e-7,1.4e-4) | (3.2e-8,3.1e-6) |
| 8 | (1287,43757) | 0:04:13 | 0:10:23 | (5.6e-6,3.1e-4) | (2.2e-7, 1.8e-6) |
| 9 | (2002,92377) | 0:13:13 | 0:43:28 | (2.2e-4,8.4e-4) | (1.1e-6,2.9e-6) |
| 10 | (3003,184755) | 3:53:13 | 4:02:11 | (2.3e-3,4.1e-3) | (4.7e-7,4.2e-6) |

**Example 2.3.2** (Sensor Network Localization)**.**

Given a graph $G = (V, E)$ and a distance for each edge, the sensor network localization problem is to find locations of vertices so that their distances are equal to the desired ones. This problem can be formulated as follows: find a sequence of unknown vectors (*sensors*) $u_1, u_2, \ldots, u_s \in \mathbb{R}^k$ (typically $k = 1, 2, 3$, we focus on $k = 2$ in this example) such that the distances between these sensors and some other fixed vectors (*anchors*) $a_1, \ldots, a_\ell$ are equal to given distances. Recently, there is much work on solving sensor network localization by SDP techniques, like [4, 47, 68]. Given edge subsets

$$\mathcal{E}_S \subset \{(i,j) : 1 \le i < j \le s\}, \quad \mathcal{E}_A = \{(i,j) : 1 \le i \le s, 1 \le j \le \ell\},$$

for every $(i,j) \in \mathcal{E}_S$, let $d_{ij}$ be the distance between $u_i$ and $u_j$, and for every $(i,j) \in \mathcal{E}_A$, let $e_{ij}$ be the distance between $u_i$ and $a_j$. Denote $u_i = (x_{ki-k+1}, \ldots, x_{ki})$ for $i = 1, \ldots, s$. The sensor network localization problem is equivalent to finding coordinates $x_{k1}, \ldots, x_{ks}$ satisfying the equations

$$\|u_i - u_j\|_2^2 = d_{ij}^2 \quad \forall\, (i,j) \in \mathcal{E}_S, \qquad \|u_i - a_j\|_2^2 = e_{ij}^2 \quad \forall\, (i,j) \in \mathcal{E}_A.$$

It is also equivalent to the quartic polynomial optimization problem

$$\min_{u_1,\ldots,u_s} \quad \sum_{(i,j)\in\mathcal{E}_S} \left(\|u_i - u_j\|_2^2 - d_{ij}^2\right)^2 + \sum_{(i,j)\in\mathcal{E}_A} \left(\|u_i - a_j\|_2^2 - e_{ij}^2\right)^2. \tag{2.30}$$

Typically, it is large scale. We use `SDPNAL` to solve its SDP relaxation (2.2)-(2.3). To test its performance, we randomly generate sensors $u_1, \ldots, u_s$ from the square $[-0.5, 0.5] \times [-0.5, 0.5]$. Fix four anchors as $(\pm 0.45, \pm 0.45)$. For each pair $(i,j)$, select it to $\mathcal{E}_S$ with probability 0.6 and to $\mathcal{E}_A$ with probability 0.3. For each instance, we test 15 times. Then compute each distance $d_{ij}$ and $e_{ij}$. After the SDP

**Table 2.6**: Computational results for sensor network localization problems.

| #sensor | time(min,max) | | RMSD(min,max) | errsdp(min,max) |
|---|---|---|---|---|
| 15 | 0:00:24 | 0:02:02 | (8.1e-6, 1.4e-4) | (1.1e-7, 1.6e-6) |
| 20 | 0:02:04 | 0:09:12 | (1.5e-5, 1.5e-4) | (2.9e-7, 2.0e-6) |
| 25 | 0:14:18 | 1:12:21 | (4.3e-5, 2.2e-4) | (2.4e-7, 1.6e-6) |
| 30 | 1:22:18 | 5:51:36 | (2.3e-5, 2.7e-3) | (9.2e-8, 5.3e-4) |
| 35 | 09:59:35 | 27:08:37 | (1.3e-3, 2.2e-3) | (6.5e-6, 6.5e-4) |
| 40 | 48:33:59 | 61:19:58 | (1.2e-3, 2.7e-3) | (2.2e-3, 4.0e-3) |

relaxation is solved, we use $Z^*(2:n+1,1)$ as a starting point and apply function "*lsqnonlin*" (in Matlab Toolbox) to get a local solution $(\hat{u}_1,\ldots,\hat{u}_s)$ of (2.30) (we use the same technique as in [68]). The errors of computed locations are measured by the Root Mean Square Distance RMSD $= (\frac{1}{s}\Sigma_{i=1}^{s}\|\hat{u}_i - u_i^*\|^2)^{1/2}$, as used in [4].

The computational results are shown in Table 2.6. We can see that the SDP relaxation of (2.30) is solved reasonably well. In general instances, FEC (2.5) or FTC (2.6) is not satisfied, so we can only get a local minimizer of (2.30) by using the technique from [68]. The true locations of sensors are found with small errors. Possible reasons for FTC (2.6) fails might be: the SDP relaxation was not solved accurately enough, or it is not exact for (2.30). □

**Example 2.3.3** (Random Constrained Polynomial Optimization)**.**

**Table 2.7**: Computational results for random Example 2.3.3

| $(n,2d)$ | time(min,max) | | errsol(min,med,max) | errsdp(min,med,max) |
|---|---|---|---|---|
| (30,4) | 0:00:28 | 0:02:47 | (5.6e-8, 1.3e-6, 6.9e-6) | (1.3e-7, 8.1e-7, 2.9e-6) |
| (40,4) | 0:03:35 | 0:10:32 | (8.8e-8, 1.8e-6, 9.5e-6) | (2.2e-7, 1.0e-6, 4.5e-6) |
| (50,4) | 0:20:34 | 0:24:59 | (5.7e-6, 5.6e-6, 7.0e-6) | (2.7e-6, 2.8e-6, 3.4e-6) |
| (60,4) | 0:35:02 | 1:20:38 | (1.5e-7, 3.5e-6, 2.5e-5) | (1.7e-7, 1.7e-6, 1.2e-5) |
| (20,6) | 0:36:31 | 0:49:17 | (8.5e-7, 2.7e-6, 4.4e-6) | (5.8e-7, 1.3e-6, 2.7e-6) |
| (12,8) | 0:27:11 | 0:59:30 | (5.5e-7, 2.8e-6, 9.0e-6) | (9.0e-7, 1.3e-6, 4.2e-6) |
| (9,10) | 0:16:31 | 0:40:53 | (2.6e-7, 3.3e-6, 1.4e-5) | (2.7e-7, 1.6e-6, 6.3e-6) |
| (80,4) | 10:52:30 | 15:57:30 | (5.3e-6, 5.5e-6, 2.2e-1) | (2.6e-6, 2.6e-6, 2.7e-3) |
| (25,6) | 10:38:04 | 12:57:59 | (5.9e-3, 6.6e-3, 1.4e-2) | (3.6e-3, 5.8e-3, 6.1e-3) |

We test the performance of Algorithm 1.3.5 (implemented by `SDPNAL` [83])

in minimizing polynomials over the unit ball. Generate $f(x)$ randomly as

$$f(x) = \sum_{\alpha \in \mathbb{N}^n : |\alpha| \le 2d} f_\alpha x^\alpha,$$

where the coefficients $f_\alpha$ are Gaussian random variables. We solve the SDP relaxation (2.11)-(2.12) by `SDPNAL`. The cases of degrees $4, 6, 8, 10$ are tested. For each instance, we test 10 times. The computational results are shown in Table 2.7. When $(n, 2d) = (80, 4)$ or $(25, 6)$, the SDP relaxations are not solved very well sometimes. This is probably because of the incurred ill-conditioning. In all the other cases, the SDP relaxations are solved quite well, and accurate global minimizers of polynomials over the unit ball are found. □

Chapter 2 Sections 2.1 and 2.3, in full, are reprint of the material as it appears in the article "Regularization Methods for SDP relaxations in Large Scale Polynomial Optimization" by Jiawang Nie and Li Wang, in SIAM Journal on Optimization, Volume 22, No.2(2012). The dissertation author was one of the authors of this paper.

Chapter 2 Section 2.2, in full, is a reprint of the material as it appears in the article "Minimizing Rational Functions by Exact Jacobian SDP relaxation Applicable to Finite Singularities" by Feng Guo, Li Wang and Guangming Zhou, in the Journal of Global Optimization in volume 58, No.2(2014). The dissertation author was one of the authors of this paper.

# Chapter 3

# Minimizing Rational Functions

## 3.1 Introduction

Consider the problem of minimizing a rational function

$$\begin{cases} r^* := \min\limits_{x\in\mathbb{R}^n} \dfrac{p(x)}{q(x)} \\ \qquad\quad \text{s.t. } h_i(x) = 0, \quad i \in [m_1], \\ \qquad\qquad g_j(x) \geq 0, \quad j \in [m_2], \end{cases} \tag{3.1}$$

where $p(x), q(x), h_i(x), g_j(x) \in \mathbb{R}[x] := \mathbb{R}[x_1, \ldots, x_n]$. As a special case, when $\deg(q) = 0$, (3.1) becomes a multivariate polynomial optimization which is NP-hard as discussed in Chapter 1 and Chapter 2.

Some approaches using sum-of-squares relaxation to solve (3.1) are proposed in [29, 54] and the core idea therein is in the following. Let $S$ be the feasible set of (3.1). Suppose that $r^* > -\infty$, and $q(x)$ is nonnegative on $S$ (otherwise replace $\frac{p(x)}{q(x)}$ by $\frac{p(x)q(x)}{q^2(x)}$), then $\gamma \in \mathbb{R}$ is a lower bound of $r^*$ if and only if $p(x) - \gamma q(x) \geq 0$ on $S$. Thus the problem (3.1) can be reformulated as maximizing $\gamma$ such that $p(x) - \gamma q(x)$ is nonnegative on $S$, which is related to the representation of a nonnegative polynomial on a semialgebraic set. As is well-known, a univariate polynomial is nonnegative on $\mathbb{R}$ if and only if it is SOS [65] which can be efficiently determined by solving a semidefinite program [61, 62]. However, when $n > 1$, due to the fact that a nonnegative multivariate polynomial might not be an SOS [65], the problem (3.1) becomes very hard even if there are no constraints.

Let $M(S)$ be the quadratic module generated by the defining polynomials of $S$, and $P(S)$ be the preordering. If $S$ in (3.1) is archimedean or compact, we can apply Putinar's Positivstellensatz (Theorem 1.2.14) or Schmüdgen's Positivstellensatz (Theorem 1.2.15) to maximize $\gamma$ such that $p(x) - \gamma q(x)$ belongs to $M(S)$ or $P(S)$.

In this chapter, we present a different way to obtain the minimum $r^*$. Given a polynomial $f \in \mathbb{R}[x]$, let $\tilde{x} = (x_0, x_1, \ldots, x_n) \in \mathbb{R}^{n+1}$ and $f^{\text{hom}}$ be the homogenization of $f$, i.e. $f^{\text{hom}}(\tilde{x}) = x_0^{\deg(f)} f(x/x_0)$. We reformulate the minimization of (3.1) by the technique of homogenization as the following polynomial optimization

$$\begin{cases} s^* := \min\limits_{\tilde{x} \in \mathbb{R}^{n+1}} \tilde{p}(\tilde{x}) \\ \qquad \text{s.t. } h_i^{\text{hom}}(\tilde{x}) = 0, \quad i \in [m_1], \\ \qquad\quad g_j^{\text{hom}}(\tilde{x}) \geq 0, \quad j \in [m_2], \\ \qquad\quad \tilde{q}(\tilde{x}) = 1, \quad x_0 \geq 0, \end{cases} \tag{3.2}$$

where $\tilde{p}(\tilde{x}) := x_0^{\max\{\deg(p), \deg(q)\}} p(x/x_0)$ and $\tilde{q}(\tilde{x}) := x_0^{\max\{\deg(p), \deg(q)\}} q(x/x_0)$. We show that these two problems are equivalent under some general conditions. As a special case, they are always equivalent if there are no constraints in (3.1). The relations between the achievabilities of $r^*$ and $s^*$ are discussed.

Therefore, the problem of solving (3.1) becomes to efficiently solving problem (3.2). Let $\tilde{S}$ be the feasible set of problem (3.2). If $\tilde{S}$ is Archimedean, the standard Lasserre's SDP relaxation presented in Section 2.1 can be applied to solve problem (3.2) efficiently. When the optimum of problem (3.2) is an asymptotic value, we refer to the approaches proposed in [15, 71, 78, 79]. However, the finite convergence of the above methods is unknown which means that we need to solve a big number of SDPs until the convergence is met. Jacobian SDP relaxation presented in Section 2.2 is exact under some generic assumptions on the feasible set and it has finite convergence guarantee under weaker assumptions, so in this chapter we employ the Jacobian SDP relaxation to solve (3.2).

Another possible and natural reformulation of (3.1) is

$$\begin{cases} \bar{s}^* := \min\limits_{x\in\mathbb{R}^n, y\in\mathbb{R}} p(x)y \\ \qquad\qquad \text{s.t. } h_i(x) = 0, \quad i \in [m_1], \\ \qquad\qquad\quad g_j(x) \geq 0, \quad j \in [m_2], \\ \qquad\qquad\quad q(x)y = 1. \end{cases} \tag{3.3}$$

Clearly, if $r^*$ is achievable in (3.1), then (3.3) is equivalent to (3.1) and we always have $r^* = \bar{s}^*$. One might ask why we solve (3.3) instead of (3.2). The reason is that when we employ Jacobian SDP relaxation [51] to solve (3.2) or (3.3), we need to assume that the optimum is achievable. Actually, $s^*$ in (3.2) is more likely to be achievable than $\bar{s}^*$ in (3.3). To see this, note that when $r^*$ is not achievable, $\bar{s}^*$ can not be reached either. However, $s^*$ might still be achievable when $r^*$ is not. Some sufficient conditions are given in Theorem 3.2.8 and they are not necessary (see Example 3.5.2 and 3.5.5). For a simple example, consider the problem

$$\min_{x_1\in\mathbb{R}} \frac{1}{x_1^2+1}.$$

Obviously, $r^* = \bar{s}^* = 0$ and they are not achievable. However, we can reformulate it as

$$\begin{cases} s^* := \min\limits_{x_0, x_1\in\mathbb{R}} x_0^2 \\ \qquad\qquad \text{s.t. } x_1^2 + x_0^2 = 1. \end{cases}$$

Then $s^* = 0$ and we have two minimizers $(0, \pm 1)$ which verify that $r^*$ is not achievable by (c) in Theorem 3.2.8.

## 3.2 Equivalent Reformulation by Homogenization

In this section, we reformulate the minimization of (3.1) as polynomial optimization (3.2) by the technique of homogenization and investigate the relations between the achievabilities of the optima of these two problems.

Given a polynomial $f \in \mathbb{R}[x]$, let $\tilde{x} = (x_0, x) = (x_0, x_1, \ldots, x_n) \in \mathbb{R}^{n+1}$ and $f^{\text{hom}}(\tilde{x})$ be the homogenization of $f$, i.e. $f^{\text{hom}}(\tilde{x}) = x_0^{\deg(f)} f(x/x_0)$. We define the

following sets:

$$\begin{aligned} S &:= \{x \in \mathbb{R}^n \mid h_i(x) = 0,\ g_j(x) \geq 0,\ i \in [m_1],\ j \in [m_2]\}, \\ \widetilde{S}_0 &:= \left\{\tilde{x} \in \mathbb{R}^{n+1} \mid h_i^{\text{hom}}(\tilde{x}) = 0,\ g_j^{\text{hom}}(\tilde{x}) \geq 0,\ x_0 > 0,\ i \in [m_1],\ j \in [m_2]\right\}, \\ \widetilde{S} &:= \left\{\tilde{x} \in \mathbb{R}^{n+1} \mid h_i^{\text{hom}}(\tilde{x}) = 0,\ g_j^{\text{hom}}(\tilde{x}) \geq 0,\ x_0 \geq 0,\ i \in [m_1],\ j \in [m_2]\right\}. \end{aligned} \tag{3.4}$$

Let $\mathsf{closure}(\tilde{S}_0)$ be the closure of $\tilde{S}_0$ in $\mathbb{R}^{n+1}$. From the above definition, we immediately have

**Proposition 3.2.1.** *$f(x) \geq 0$ on $S$ if and only if $f^{\text{hom}}(\tilde{x}) \geq 0$ on $\mathsf{closure}(\tilde{S}_0)$.*

*Proof.* We first prove the "if" part. Suppose $f^{\text{hom}}(\tilde{x}) \geq 0$ on $\mathsf{closure}(\tilde{S}_0)$. If there exists a point $u \in S$ such that $f(u) < 0$, then $(1, u) \in \widetilde{S}_0$. Thus $f^{\text{hom}}(1, u) = f(u) < 0$ which is a contradiction.

Next we prove the "only if" part. Suppose $f(x) \geq 0$ on $S$ and consider a point $(u_0, u) \in \mathbb{R}^{n+1}$ in $\mathsf{closure}(\tilde{S}_0)$. There exists a sequence $\{(u_{k,0}, u_k)\} \in \widetilde{S}_0$ such that $\lim_{k\to\infty}(u_{k,0}, u_k) = (u_0, u)$. Since $u_{k,0} > 0$ for all $k \in \mathbb{N}$, we consider the sequence $\{u_k/u_{k,0}\}$. For $i = 1, \ldots, m_1$ and $j = 1, \ldots, m_2$, we have $h_i(u_k/u_{k,0}) = h_i^{\text{hom}}(u_{k,0}, u_k)/(u_{k,0})^{\deg(h_i)} = 0$ and $g_j(u_k/u_{k,0}) = g_j^{\text{hom}}(u_{k,0}, u_k)/(u_{k,0})^{\deg(g_j)} \geq 0$. It implies that $\{u_k/u_{k,0}\} \in S$. Thus

$$f^{\text{hom}}(u_0, u) = \lim_{k\to\infty} f^{\text{hom}}(u_{k,0}, u_k) = \lim_{k\to\infty} u_{k,0}^{\deg(f)} f(u_k/u_{k,0}) \geq 0,$$

which concludes the proof. □

Let $d = \max\{\deg(p), \deg(q)\}$, and define

$$\tilde{p}(\tilde{x}) = x_0^d p(x/x_0) \quad \text{and} \quad \tilde{q}(\tilde{x}) = x_0^d q(x/x_0).$$

We reformulate the minimization problem (3.1) as the following constrained polynomial optimization:

$$\begin{cases} s^* := \min\limits_{\tilde{x}\in\mathbb{R}^{n+1}} \tilde{p}(\tilde{x}) \\ \qquad \text{s.t. } h_i^{\text{hom}}(\tilde{x}) = 0, \quad i \in [m_1], \\ \qquad\qquad g_j^{\text{hom}}(\tilde{x}) \geq 0, \quad j \in [m_2], \\ \qquad\qquad \tilde{q}(\tilde{x}) = 1, \quad x_0 \geq 0, \end{cases} \tag{3.5}$$

We now investigate the relations between $r^*$ and $s^*$.

**Assumption 3.2.2.** *If $r^*$ is achievable, then there exist a minimizer $x^*$ of (3.1) and a neighborhood $\mathcal{O}$ of $x^*$ such that $q(x) > 0$ for every $x \in \mathcal{O} \cap S$; if $r^*$ is not achievable, then $q(x) > 0$ for every $x \in S$ with Euclidean norm $\|x\|$ sufficiently large.*

If Assumption 3.2.2 does not hold, then we can replace $\frac{p(x)}{q(x)}$ by $\frac{p(x)q(x)}{q(x)^2}$. Note that we do not assume $q(x)$ is nonnegative on the whole feasible set $S$ as in [29, 54].

**Definition 3.2.3.** ([51]) If there exists a point $0 \neq (0, u) \in \tilde{S}$ but $(0, u) \notin \mathsf{closure}(\tilde{S}_0)$, then we say $S$ is not closed at $\infty$; otherwise, we say $S$ is closed at $\infty$.

**Theorem 3.2.4.** *It always holds that $s^* \leq r^*$, and the equality holds if one of the following conditions is satisfied:*

(a) *$S$ is closed at $\infty$;*

(b) $\deg(p) > \deg(q)$*;*

(c) *$s^*$ is achievable and $x_0^* > 0$ for at least one of its minimizers $\tilde{x}^* = (x_0^*, x^*)$.*

*Proof.* We first show that $s^* \leq r^*$. For any $u \in S$ in a neighborhood of a minimizer of (3.1) or with sufficient large Euclidean norm if $r^*$ is not achievable, if $\frac{p(x)}{q(x)}$ is defined at $u$, then $q(u) > 0$ by the Assumption 3.2.2. Let $t = q(u)^{1/d} = \tilde{q}(1, u)^{1/d}$. We have $\tilde{q}(1/t, u/t) = 1$ and $(1/t, u/t) \in \widetilde{S}$, so

$$\frac{p(u)}{q(u)} = \frac{\tilde{p}(1, u)}{\tilde{q}(1, u)} = \frac{\tilde{p}(1/t, u/t)}{\tilde{q}(1/t, u/t)} = \tilde{p}(1/t, u/t) \geq s^*,$$

then we have $s^* \leq r^*$. Therefore, to show $r^* = s^*$, we only need to show $r^* \leq s^*$.

(a) For any feasible point $(u_0, u)$ of (3.2), i.e., $(u_0, u) \in \widetilde{S}$ and $\tilde{q}(u_0, u) = 1$, since $S$ is closed at $\infty$, there exists a sequence $\{(u_{k,0}, u_k)\}$ in $\widetilde{S}$ such that $u_{k,0} > 0$ for any $k \in \mathbb{N}$ and $\lim\limits_{k\to\infty}(u_{k,0}, u_k) = (u_0, u)$. Due to the continuity of $\tilde{q}$, $\lim\limits_{k\to\infty} \tilde{q}(u_{k,0}, u_k) = 1$. Hence, we can always assume that for any $k \in \mathbb{N}$, $\tilde{q}(u_{k,0}, u_k) > 0$. For each $k \in \mathbb{N}$, let $t_k = \tilde{q}(u_{k,0}, u_k)^{1/d}$ and consider the sequence $\{(u_{k,0}/t_k, u_k/t_k)\}$. We have $\lim\limits_{k\to\infty}(u_{k,0}/t_k, u_k/t_k) = (u_0, u_k)$ and $\tilde{q}(u_{k,0}/t_k, u_k/t_k) = 1$. For $i = 1, \ldots, m_1$,

$j = 1, \ldots, m_2$,

$$0 = \frac{1}{t_k^{\deg(h_i)}} h_i^{\hom}(u_{k,0}, u_k) = h_i^{\hom}(u_{k,0}/t_k, u_k/t_k) = \frac{1}{t_k^{\deg(h_i)}} u_{k,0}^{\deg(h_i)} h_i(u_k/u_{k,0}),$$
$$0 \leq \frac{1}{t_k^{\deg(g_j)}} g_j^{\hom}(u_{k,0}, u_k) = g_j^{\hom}(u_{k,0}/t_k, u_k/t_k) = \frac{1}{t_k^{\deg(g_j)}} u_{k,0}^{\deg(g_j)} g_j(u_k/u_{k,0}),$$

which imply $(u_{k,0}/t_k, u_k/t_k) \in \widetilde{S}$ and $u_k/u_{k,0} \in S$ for all $k$. Hence

$$\tilde{p}(u_{k,0}/t_k, u_k/t_k) = \frac{\tilde{p}(u_{k,0}/t_k, u_k/t_k)}{\tilde{q}(u_{k,0}/t_k, u_k/t_k)} = \frac{p(u_k/u_{k,0})}{q(u_k/u_{k,0})} \geq r^*$$

and $\tilde{p}(u_0, u) = \lim_{k\to\infty} \tilde{p}(u_{k,0}/t_k, u_k/t_k) \geq r^*$ which means $r^* \leq s^*$.

(b) If $\deg(p) > \deg(q)$, then $x_0$ divides $\tilde{q}(\tilde{x})$. By $\tilde{q}(\tilde{x}) = 1$, we have $u_0 > 0$ for any feasible point $(u_0, u)$ of (3.2) and it is easy to see that $u/u_0 \in S$, then

$$\tilde{p}(u_0, u) = \frac{\tilde{p}(u_0, u)}{\tilde{q}(u_0, u)} = \frac{\tilde{p}(1, u/u_0)}{\tilde{q}(1, u/u_0)} = \frac{p(u/u_0)}{q(u/u_0)} \geq r^*,$$

which means $r^* \leq s^*$.

(c) Since $x_0^* > 0$, we have $x^*/x_0^* \in S$ and

$$s^* = \tilde{p}(x_0^*, x^*) = \frac{\tilde{p}(x_0^*, x^*)}{\tilde{q}(x_0^*, x^*)} = \frac{p(x^*/x_0^*)}{q(x^*/x_0^*)} \geq r^*,$$

which implies $r^* = s^*$. □

If there are no constraints in (3.1), then $S = \mathbb{R}^n$ is closed at $\infty$ since $\{\tilde{x} \in \mathbb{R}^{n+1} | x_0 \geq 0\}$ is the closure of $\{\tilde{x} \in \mathbb{R}^{n+1} | x_0 > 0\}$. Therefore,

**Corollary 3.2.5.** *If $S = \mathbb{R}^n$, then $r^* = s^*$.*

**Remark 3.2.6.** If $S = \mathbb{R}^n$, we can remove $x_0 \geq 0$ in (3.2). In fact, if there are no constraints, according to the proof of Part $(a)$ in Theorem 3.2.4, we only need $u_{k,0} \neq 0$ to get the same result. Therefore, the global minimization

$$r^* := \min_{x \in \mathbb{R}^n} \frac{p(x)}{q(x)}$$

is equivalent to

$$\begin{cases} s^* := \min\limits_{\tilde{x} \in \mathbb{R}^{n+1}} \tilde{p}(\tilde{x}) \\ \qquad\quad s.t.\ \tilde{q}(\tilde{x}) = 1. \end{cases} \tag{3.6}$$

We would like to point out that not every $S$ is closed at $\infty$ and $s^*$ might be strictly smaller than $r^*$ in this case. For example,

**Example 3.2.7.**

$$\begin{cases} r^* := \min\limits_{x_1,x_2\in\mathbb{R}} \dfrac{x_1}{(x_1-x_2)^2} \\ \qquad\qquad \text{s.t. } x_1^2(x_1-x_2)=1, \\ \qquad\qquad\quad x_1-1\geq 0. \end{cases} \tag{3.7}$$

Clearly, we have $r^*=1$. However, [49, Example 5.2 (i)] shows that the set

$$\{(x_1,x_2)\in\mathbb{R}^2 \mid x_1^2(x_1-x_2)-1=0\}$$

is not closed at $\infty$. Actually,

$$S:=\{(x_1,x_2)\in\mathbb{R}^2 \mid x_1^2(x_1-x_2)-1=0, x_1-1\geq 0\}$$

is not closed at $\infty$, either. To see it, we have

$$\widetilde{S}:=\{(x_0,x_1,x_2)\in\mathbb{R}^3 \mid x_1^2(x_1-x_2)-x_0^3=0, x_1-x_0\geq 0, x_0\geq 0\}.$$

Consider the point $(0,0,1)\in\widetilde{S}$. Suppose that there exists a sequence $\{(x_{k,0},x_{k,1},x_{k,2})\}$ in $\widetilde{S}$ such that $\lim\limits_{k\to\infty}(x_{k,0},x_{k,1},x_{k,2})=(0,0,1)$ and $x_{k,0}>0$ for all $k\in\mathbb{N}$. Then for $0<\varepsilon<1/2$, there exists $N\in\mathbb{N}$ such that for any $k>N$, we have

$$0<x_{k,0}<\varepsilon,\ |x_{k,1}|<\varepsilon,\ |x_{k,2}-1|<\varepsilon.$$

Thus

$$0<x_{k,0}^3=x_{k,1}^2(x_{k,1}-x_{k,2})<x_{k,1}^2(\varepsilon-1+\varepsilon)<0$$

which is a contradiction. Therefore, $S$ is not closed at $\infty$ and if we reformulate (3.7) by homogenization as the following problem

$$\begin{cases} s^* := \min\limits_{x_0,x_1,x_2\in\mathbb{R}} x_0x_1 \\ \qquad\qquad \text{s.t. } (x_1-x_2)^2-1=x_1^2(x_1-x_2)-x_0^3=0, \\ \qquad\qquad\quad x_1-x_0\geq 0, x_0\geq 0, \end{cases}$$

then we have $s^*=0<r^*$.

Let

$$\widehat{S} := \{x \in \mathbb{R}^n \mid \hat{h}_i(x) = 0,\ \hat{g}_j(x) \geq 0,\ i = 1, \ldots, m_1,\ j = 1, \ldots, m_2\}$$

where $\hat{h}_i$ and $\hat{g}_j$ denote the homogeneous parts of the highest degree of $h_i$ and $g_j$, respectively. Denote $p_d(x)$ and $q_d(x)$ the homogeneous parts of degree $d$ of $p(x)$ and $q(x)$, respectively.

**Theorem 3.2.8.** *If one of the conditions in Theorem 3.2.4 holds, then the following properties hold.*

(a) *$r^*$ is achievable if and only if $s^*$ is achievable at a minimizer $\tilde{x}^* = (x_0^*, x^*)$ with $x_0^* \neq 0$;*

(b) *If neither $p(x)$ and $q(x)$ have real common roots in $S$, nor $p_d(x)$ and $q_d(x)$ have real nonzero common roots in $\widehat{S}$, then $s^*$ is achievable.*

(c) *If $s^*$ is achievable and $x_0^* = 0$ for all minimizers $\tilde{x}^* = (x_0^*, x^*)$ of (3.2), then $r^*$ is not achievable. For each minimizer $\tilde{x}^* = (0, x^*)$ of (3.2), if there exists a sequence $\{\tilde{x}_k\} = \{(x_{k,0}, x_k)\}$ in $\widetilde{S}$ such that $\lim\limits_{k\to\infty} \tilde{x}_k = \tilde{x}^*$ and $x_{k,0} > 0$ for all $k \in \mathbb{N}$, then $\lim\limits_{k\to\infty} \frac{p(x_k/x_{k,0})}{q(x_k/x_{k,0})} = r^*$.*

*Proof.* If one of the conditions in Theorem 3.2.4 holds, we have $r^* = s^*$.

(a) Let $x^*$ be a minimizer of (3.1) such that $t = \tilde{q}(1, x^*)^{1/d} = q(x^*)^{1/d} > 0$ by the Assumption 3.2.2. It is easy to verify that $(1/t, x^*/t) \in \widetilde{S}$ and $\tilde{q}(1/t, x^*/t) = 1$. We have $\tilde{p}(1/t, x^*/t) = r^* = s^*$ which means $(1/t, x^*/t)$ is a minimizer of (3.2). If $s^*$ is achieved at $\tilde{x}^* = (x_0^*, x^*) \in \widetilde{S}$ with $x_0^* > 0$, then $r^*$ is achieved at $x^*/x_0^* \in S$.

(b) To the contrary, we assume that $s^*$ is not achievable. Then there exists a sequence $\{\tilde{x}_k\}$ in $\widetilde{S}$ such that $\lim\limits_{k\to\infty} ||\tilde{x}_k||_2 = \infty$, $\lim\limits_{k\to\infty} \tilde{p}(\tilde{x}_k) = s^*$ and for all $k \in \mathbb{N}$, $\tilde{q}(\tilde{x}_k) = 1$. Consider the bounded sequence $\{\tilde{x}_k/||\tilde{x}_k||_2\} \subseteq \widetilde{S}$. By Bolzano-Weierstrass Theorem, there exists a subsequence $\{\tilde{x}_{k_j}/||\tilde{x}_{k_j}||_2\}$ such that $\lim\limits_{j\to\infty} \tilde{x}_{k_j}/||\tilde{x}_{k_j}||_2 = \tilde{y}$ for some nonzero $\tilde{y} = (y_0, y) \in \widetilde{S}$ since $\widetilde{S}$ is closed. Let $\tilde{p}(\tilde{x}_{k_j}) = s_{k_j}$, then $\lim\limits_{j\to\infty} s_{k_j} = s^*$. Since $\tilde{p}(\tilde{x}_{k_j}) = (||x_{k_j}||_2)^d \tilde{p}(\tilde{x}_{k_j}/||\tilde{x}_{k_j}||_2)$ and $\lim\limits_{j\to\infty} ||\tilde{x}_{k_j}||_2 = \infty$, we have $\tilde{p}(\tilde{y}) = \lim\limits_{j\to\infty} \tilde{p}(\tilde{x}_{k_j}/||\tilde{x}_{k_j}||_2) = 0$. Similarly, we can prove $\tilde{q}(\tilde{y}) = \lim\limits_{j\to\infty} \tilde{q}(\tilde{x}_{k_j}/||\tilde{x}_{k_j}||_2) = 0$. Thus $\tilde{p}(\tilde{x})$ and $\tilde{q}(\tilde{x})$ have real nonzero common root

$\tilde{y}$ on unit sphere $S^{n+1}$. We have $y_0 = 0$, otherwise $y/y_0$ is a real common root of $p(x)$ and $q(x)$ in $S$. Therefore $0 = \tilde{p}(\tilde{y}) = p_d(y),\ 0 = \tilde{q}(\tilde{y}) = q_d(y)$, $0 = h_i^{\text{hom}}(\tilde{y}) = \hat{h}_i(y)$ and $0 \leq g_j^{\text{hom}}(\tilde{y}) = \hat{g}_j(y)$, i.e. $p_d(x)$ and $q_d(x)$ have real nonzero common root $y$ in $\widehat{S}$ which is a contradiction.

(c) By (a), if $x_0^* = 0$ for all minimizers of (3.2), $r^*$ is not achievable. Suppose $\tilde{x}^* = (0, x^*)$ is a minimizer of (3.2) and there exists a sequence $\{\tilde{x}_k\} = \{(x_{k,0}, x_k)\}$ in $\widetilde{S}$ such that $\lim_{k\to\infty} \tilde{x}_k = \tilde{x}^*$ and $x_{k,0} > 0$. Then for each $k \in \mathbb{N}$, $x_k/x_{k,0} \in S$. Since $\tilde{p}$ and $\tilde{q}$ are continuous, $\lim_{k\to\infty} \tilde{p}(x_{k,0}, x_k) = s^*$ and $\lim_{k\to\infty} \tilde{q}(x_{k,0}, x_k) = 1$. Therefore,

$$\lim_{k\to\infty} \frac{p(x_k/x_{k,0})}{q(x_k/x_{k,0})} = \lim_{k\to\infty} \frac{\tilde{p}(x_{k,0}, x_k)}{\tilde{q}(x_{k,0}, x_k)} = s^* = r^*.$$

Here completes the proof. □

In this section, we reformulate the minimization of (3.1) as the polynomial optimization (3.2) by homogenization. Suppose $S$ is closed at $\infty$ which is always true when $S = \mathbb{R}^n$, then $r^* = s^*$. If we have no information about whether $S$ is closed at $\infty$ or not, but $\deg(p) > \deg(q)$ or $s^*$ is achieved at $(x_0^*, x^*)$ with $x_0^* \neq 0$, we still have $r^* = s^*$. Optimum $s^*$ in (3.2) is achievable under some sufficient conditions (a) and (b) in Proposition 3.2.8. If $r^* = s^*$, the relations between the achievabilities of $r^*$ and $s^*$ are discussed. In next section, we will discuss the assumption that $S$ is closed at $\infty$ is a generic condition.

## 3.3 On the Generality of Closedness at $\infty$

Although we have counter example in Example 3.2.7, we next show that a given set $S$ in (3.4) is generically closed at $\infty$. Therefore, if the constraints in (3.1) are generic, (3.1) and (3.2) are equivalent.

Let us first review some elementary definitions about *resultants* and *discriminants*. More details can be found in [13, 49, 51]. Let $f_1, \ldots, f_n$ be homogeneous polynomials in $x = (x_1, \ldots, x_n)$. The resultant $Res(f_1, \ldots, f_n)$ is a polynomial in the coefficients of $f_1, \ldots, f_n$ satisfying

$$Res(f_1, \ldots, f_n) = 0 \quad \Leftrightarrow \quad \exists\ 0 \neq u \in \mathbb{C}^n,\ f_1(u) = \cdots = f_n(u) = 0.$$

Let $f_1, \ldots, f_m$ be homogenous polynomials with $m < n$. The discriminant for $f_1, \ldots, f_m$ is denoted by $\Delta(f_1, \ldots, f_m)$, which is a polynomial in the coefficients of $f_1, \ldots, f_m$ such that

$$\Delta(f_1, \ldots, f_m) = 0$$

if and only if the polynomial system

$$f_1(x) = \cdots = f_m(x) = 0$$

has a solution $0 \neq u \in \mathbb{C}^n$ such that the Jacobian matrix of $f_1, \ldots, f_m$ does not have full rank.

Suppose $S$ is not closed at $\infty$, then by definition there exists $(0, u) \in \tilde{S} \backslash \mathsf{closure}(\tilde{S}_0)$ where $u \in \mathbb{R}^n$. Let $J(u) := \{j \in [m_2] \mid g_j^{\hom}(0, u) = 0\}$. Then $g_j^{\hom}(0, u) > 0$ for all $j \in [m_2] \backslash J(u)$. We have the cardinality $m_1 + |J(u)| \geq 1$, otherwise, $(0, u)$ is an interior point of $\tilde{S}$ and $(0, u) \in \mathsf{closure}(\tilde{S}_0)$. Let

$$V(u) := \{\tilde{x} \in \mathbb{R}^{n+1} \mid h_i^{\hom}(\tilde{x}) = 0,\ g_j^{\hom}(\tilde{x}) = 0,\ i \in [m_1],\ j \in J(u)\}.$$

For any $\delta > 0$, let

$$B((0, u), \delta) = \{(x_0, x) \in \mathbb{R}^{n+1} \mid \|(x_0, x) - (0, u)\| \leq \delta\}.$$

**Lemma 3.3.1.** *Suppose $S$ is not closed at $\infty$, then there exists $\delta > 0$ such that for all $(x_0, x) \in B((0, u), \delta) \cap V(u)$, we have $x_0 \leq 0$.*

*Proof.* Suppose that $\delta$ does not exist. Consider a sequence $\{\delta_k\}$ with $\delta_k > 0$ and $\lim_{k \to \infty} \delta_k = 0$. Then for each $k$, there exists a point $(u_{k,0}, u_k) \in B((0, u), \delta_k) \cap V(u)$ such that $u_{k,0} > 0$. By the continuity, there exists $N$ such that for all $k \geq N$, $g_j^{\hom}(u_{k,0}, u_k) > 0$ for each $j \in [m_2] \setminus J(u)$ which implies $(u_{k,0}, u_k) \in \tilde{S}$ for all $k \geq N$ and $(0, u) \in \mathsf{closure}(\tilde{S}_0)$. The contradiction follows. □

Let $J(u) = \{j_1, \cdots, j_l\}$ and

$$A(u) := \begin{bmatrix} \frac{\partial h_1^{\hom}}{\partial x_1}(0, u) & \cdots & \frac{\partial h_1^{\hom}}{\partial x_n}(0, u) \\ \vdots & \vdots & \vdots \\ \frac{\partial h_{m_1}^{\hom}}{\partial x_1}(0, u) & \cdots & \frac{\partial h_{m_1}^{\hom}}{\partial x_n}(0, u) \\ \frac{\partial g_{j_1}^{\hom}}{\partial x_1}(0, u) & \cdots & \frac{\partial g_{j_1}^{\hom}}{\partial x_n}(0, u) \\ \vdots & \vdots & \vdots \\ \frac{\partial g_{j_l}^{\hom}}{\partial x_1}(0, u) & \cdots & \frac{\partial g_{j_l}^{\hom}}{\partial x_n}(0, u) \end{bmatrix} = \begin{bmatrix} \frac{\partial \hat{h}_1}{\partial x_1}(u) & \cdots & \frac{\partial \hat{h}_1}{\partial x_n}(u) \\ \vdots & \vdots & \vdots \\ \frac{\partial \hat{h}_{m_1}}{\partial x_1}(u) & \cdots & \frac{\partial \hat{h}_{m_1}}{\partial x_n}(u) \\ \frac{\partial \hat{g}_{j_1}}{\partial x_1}(u) & \cdots & \frac{\partial \hat{g}_{j_1}}{\partial x_n}(u) \\ \vdots & \vdots & \vdots \\ \frac{\partial \hat{g}_{j_l}}{\partial x_1}(u) & \cdots & \frac{\partial \hat{g}_{j_l}}{\partial x_n}(u) \end{bmatrix},$$

where $\hat{h}_i$ and $\hat{g}_j$ denote the homogenous parts of the highest degree of $h_i$ and $g_j$, respectively. Combining Lemma 3.3.1 and Implicit Function Theorem [35, Theorem 3.3.1], we have

**Lemma 3.3.2.** *Suppose $S$ is not closed at $\infty$ and $m_1 + |J(u)| \leq n$, then rank $A(u) < m_1 + |J(u)|$.*

*Proof.* Let $m = m_1 + |J(u)|$. Suppose rank $A(u) = m$. Then there exist $m$ independent columns in $A(u)$. Without loss of generality, we assume the last $m$ columns of $A(u)$ are independent, i.e., the Jacobian determinant

$$\frac{\partial(h_1^{\text{hom}}, \cdots, h_{m_1}^{\text{hom}}, g_{j_1}^{\text{hom}}, \cdots, g_{j_l}^{\text{hom}})}{\partial(x_{n-m+1}, \cdots, x_n)}(0, u) \neq 0.$$

Partition $(0, \tilde{u})$ as $(\tilde{u}^a, \tilde{u}^b)$ where $\tilde{u}^a = (0, u_1, \cdots, u_{n-m})$, $\tilde{u}^b = (u_{n-m+1}, \cdots, u_n)$. Then by the Implicit Function Theorem, there exists an open set $W \subseteq \mathbb{R}^{n-m+1}$ containing $\tilde{u}^a$ and $k$-th continuous functions $f_1, \cdots, f_m$ on $W$ such that

$$\begin{aligned} h_i^{\text{hom}}(x_0, \ldots, x_{n-m}, f_1(\tilde{x}^a), \ldots, f_m(\tilde{x}^a)) &= 0, \ i \in [m_1], \\ g_j^{\text{hom}}(x_0, \ldots, x_{n-m}, f_1(\tilde{x}^a), \ldots, f_m(\tilde{x}^a)) &= 0, \ j \in J(u), \end{aligned}$$

for every $\tilde{x}^a \in W$. Here $\tilde{x}^a = (x_0, \cdots, x_{n-m})$. Therefore, $(\tilde{x}^a, f_1(\tilde{x}^a), \ldots, f_m(\tilde{x}^a)) \in V(u)$ for every $\tilde{x}^a \in W$. Since $W$ is open and $f_1, \cdots, f_m$ are continuous, we can choose $\tilde{x}^a$ close enough to $\tilde{u}^a$ such that $(\tilde{x}^a, f_1(\tilde{x}^a), \ldots, f_m(\tilde{x}^a)) \in B((0, u), \delta) \cap V(u)$ with $x_0 > 0$ for every $\delta > 0$, which contradicts the conclusion in Lemma 3.3.1. □

The following theorem shows that if the defining polynomials of $S$ are generic, then $S$ is closed at $\infty$.

**Theorem 3.3.3.** *Suppose $S$ is not closed at $\infty$, then*

(a) *If $m_1 + |J(u)| \geq n$, then $Res(\hat{h}_1, \cdots, \hat{h}_{m_1}, \hat{g}_{j_1}, \cdots, \hat{g}_{j_{n-m}}) = 0$ for every subset $\{j_1, \cdots, j_{n-m_1}\} \subseteq J(u)$;*

(b) *If $m_1 + |J(u)| < n$, then $\Delta(\hat{h}_1, \cdots, \hat{h}_{m_1}, \hat{g}_{j_1}, \cdots, \hat{g}_{j_l}) = 0$.*

*Proof.* Since $\hat{h}_i(u) = h_i^{\text{hom}}(0, u) = 0, \hat{g}_j(u) = g_j^{\text{hom}}(0, u) = 0$ for all $i \in [m_1], j \in J(u)$, then the conclusion in (a) is implied by the properties of resultants. If $m_1 +$

$|J(u)| < n$, then by Lemma 3.3.2, the Jacobian matrix of $(\hat{h}_1, \cdots, \hat{h}_{m_1}, \hat{g}_{j_1}, \cdots, \hat{g}_{j_l})$ does not have full rank at $u$. Hence, the conclusion in (b) follows by the properties of discriminants. □

This theorem shows that if $S$ is defined by some generic polynomials, then it is closed at $\infty$. Hence, the assumption that $S$ is closed at $\infty$ is a generic condition. Therefore, problems (3.1) and (3.2) are equivalent in general. Now the problem becomes how to efficiently solve polynomial optimization (3.2). The feasible set of the reformulated problem (3.2) may not be compact, then Lasserre's SDP relaxations may not have finite convergence. In next section, we consider to apply the Jacobian SDP relaxation [51] presented in Section 2.2 to solve the reformulated problem (3.2).

## 3.4 Using the Jacobian SDP Relaxation

In this section, we apply the Jacobian SDP relaxation discussed in Section 2.2 to solve the reformulated problem (3.2). Consider the number of new constraints added when we employ Jacobian SDP relaxation to solve (3.2). As mentioned in [51], the number of new constraints in (2.15) is exponential in the number of inequality constraints. Hence, if the number of inequality constraints is large, (2.15) becomes more difficult to solve numerically. In the following, we employ the Jacobian SDP relaxation to reformulate (3.2) as (2.15). We show that the number of the new equality constraints $\varphi_i$'s in (2.15) can be reduced due to the special inequality constraint $x_0 \geq 0$ in (3.2).

In (3.2), for convenience, let

$$h^{\rm hom}_{m_1+1}(\tilde{x}) := \tilde{q}(\tilde{x}) - 1 = 0, \quad g^{\rm hom}_{m_2+1}(\tilde{x}) := x_0 \geq 0 \quad \text{and} \quad m := \min\{m_1 + m_2 + 2, n\}.$$

Denote

$$\nabla_{\tilde{x}} := \left( \frac{\partial}{\partial x_0}, \frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_n} \right).$$

According to (2.13) and (2.2.1), we need to consider all subsets of $[m_2 + 1]$ with cardinality $\leq m - m_1 - 1$. Let $l = \min\{m - m_1 - 1, m_2\}$. We first consider the subsets

without $m_2+1$, i.e., every subset $J=\{j_1,\ldots,j_k\}\subseteq[m_2]$ with $k\le l$. Denote $h^{\rm hom}=(h_1^{\rm hom},\ldots,h_{m_1}^{\rm hom},h_{m_1+1}^{\rm hom})$ and $g_J^{\rm hom}=(g_{j_1}^{\rm hom},\ldots,g_{j_k}^{\rm hom})$. Let $\{\eta_1,\ldots,\eta_{len(J)}\}$ be the set of the defining equations for the determinantal variety

$$G_J:=\{\tilde{x}\in\mathbb{C}^{n+1}\mid \text{rank }[\nabla_{\tilde{x}}\tilde{p}\quad \nabla_{\tilde{x}}h^{\rm hom}\quad \nabla_{\tilde{x}}g_J^{\rm hom}]\le m_1+|J|+1\}.$$

For each $i=1,\cdots,len(J)$, define

$$\varphi_i^J(\tilde{x})=\eta_i\cdot\prod_{j\in J^c}g_j^{\rm hom}(\tilde{x}),\quad \text{where}\quad J^c=[m_2+1]\backslash J.$$

For every subset $J$ considered above, denote $J'=J\cup\{m_2+1\}\subseteq[m_2+1]$. It can be checked that the collection of these $J$'s and $J'$'s contains all subsets of $[m_2+1]$ with cardinality $\le m-m_1-1$ and some possible $J'$'s with cardinality $=m-m_1$ (which will happen when $n<m_1+m_2+1$).

**Case** $|J'|\le m-m_1-1$ All these $J'$'s compose of the subsets of $[m_2+1]$ containing $m_2+1$ with cardinality $\le m-m_1-1$. It is easy to see that the set of the defining equations for the determinantal variety

$$G_{J'}:=\{\tilde{x}\in\mathbb{C}^{n+1}\mid \text{rank }[\nabla_{\tilde{x}}\tilde{p}\quad \nabla_{\tilde{x}}h^{\rm hom}\quad \nabla_{\tilde{x}}g_J^{\rm hom}\quad \nabla_{\tilde{x}}x_0]\le m_1+|J|+2\}$$

is a subset of $\{\eta_1,\ldots,\eta_{len(J)}\}$. We generally suppose it to be $\{\eta_1,\ldots,\eta_{t(J)}\}$ with $t(J)<len(J)$. For $i=1,\cdots,t(J)$, define

$$\varphi_i^{J'}(\tilde{x})=\eta_i\cdot\prod_{j\in J^c}g_j^{\rm hom}(\tilde{x}),\quad \text{where}\quad J^c=[m_2+1]\backslash J.$$

**Case** $|J'|=m-m_1$ It is easy to check that $G_{J'}=\mathbb{C}^{n+1}$. Thus for convenience, we set $t(J)=0$ in this case.

Then for every subset $J\subseteq[m_2]$ with $|J|\le l$, we have

$$\varphi_i^J(\tilde{x})=\varphi_i^{J'}(\tilde{x})\cdot x_0,\quad i=1,\cdots,t(J). \tag{3.8}$$

Now consider the SDP relaxations [39] for the following polynomial optimization

$$\begin{cases} p^*:=\min\limits_{\tilde{x}\in\mathbb{R}^{n+1}}\ \tilde{p}(\tilde{x}) \\ \qquad\text{s.t. } h_1^{\rm hom}(\tilde{x})=\cdots=h_{m_1}^{\rm hom}(\tilde{x})=h_{m_1+1}^{\rm hom}(\tilde{x})=0, \\ \qquad\quad \varphi_i^J(\tilde{x})=0,\varphi_j^{J'}(\tilde{x})=0 \\ \qquad\quad (i\in[len(J)],\ j\in[t(J)],\ J\subseteq[m_2],|J|\le l), \\ \qquad\quad g_\nu^{\rm hom}(\tilde{x})\ge 0,\forall\nu\in\{0,1\}^{m_2+1}, \end{cases} \tag{3.9}$$

where $g_\nu^{\rm hom} = (g_1^{\rm hom})^{\nu_1} \cdots (g_{m_2+1}^{\rm hom})^{\nu_{m_2+1}}$. We now show that for each $J \subseteq [m_2]$ with $|J| \le l$, constraints $\varphi_1^J(\tilde{x}) = \cdots = \varphi_{t(J)}^J(\tilde{x}) = 0$ can be removed from (3.9). Consider the $N$-th order SDP relaxation (2.17) for (3.9). By (3.8) and the properties of localizing moment matrices in [42, Lemma 4.1], we have

$$L_{\varphi_j^{J'}}^{(N)}(y) = 0 \quad \text{implies} \quad L_{\varphi_j^J}^{(N)}(y) = 0, \quad j \in [t(J)],\ J \subseteq [m_2],\ |J| \le l.$$

In the dual problem (2.18), by (3.8), the truncated ideal

$$\left\{ \sum_{J\subseteq[m_2],|J|\le l} \left( \sum_{i=1}^{len(J)} \phi_i \varphi_i^J + \sum_{j=1}^{t(J)} \zeta_j \varphi_j^{J'} \right) + \sum_{k=1}^{m_1+1} \psi_k h_k^{hom} \;\middle|\; \begin{array}{l} \deg(\phi_i \varphi_i^J) \le 2N\ \forall i \\ \deg(\zeta_j \varphi_j^{J'}) \le 2N\ \forall j \\ \deg(\psi_k h_k^{hom}) \le 2N\ \forall k \end{array} \right\}.$$

agrees with

$$\left\{ \sum_{J\subseteq[m_2],|J|\le l} \left( \sum_{i=t(J)+1}^{len(J)} \phi_i \varphi_i^J + \sum_{j=1}^{t(J)} \zeta_j \varphi_j^{J'} \right) + \sum_{k=1}^{m_1+1} \psi_k h_k^{\rm hom} \;\middle|\; \begin{array}{l} \deg(\phi_i \varphi_i^J) \le 2N\ \forall i \\ \deg(\zeta_j \varphi_j^{J'}) \le 2N\ \forall j \\ \deg(\psi_k h_k^{\rm hom}) \le 2N\ \forall k \end{array} \right\}. \tag{3.10}$$

Therefore, we can remove $\varphi_1^J(\tilde{x}) = \cdots = \varphi_{t(J)}^J(\tilde{x}) = 0$ in (3.9) and improve the numerical performance in practice. Hence we consider the following optimization

$$\left\{ \begin{array}{l} p^* := \min\limits_{\tilde{x}\in\mathbb{R}^{n+1}} \tilde{p}(\tilde{x}) \\ \qquad \text{s.t. } h_1^{\rm hom}(\tilde{x}) = \cdots = h_{m_1}^{\rm hom}(\tilde{x}) = h_{m_1+1}^{\rm hom}(\tilde{x}) = 0, \\ \qquad\quad \varphi_i^J(\tilde{x}) = 0, \varphi_j^{J'}(\tilde{x}) = 0 \\ \qquad\quad (i = t(J)+1, \ldots, len(J),\ j \in [t(J)],\ J \subseteq [m_2], |J| \le l), \\ \qquad\quad g_\nu^{\rm hom}(\tilde{x}) \ge 0, \forall \nu \in \{0,1\}^{m_2+1}, \end{array} \right. \tag{3.11}$$

The $N$-th order SDP relaxation [39] for (3.11) is the SDP

$$\left\{ \begin{array}{l} p_N^{(1)} := \min\ L_{\tilde{p}}(y) \\ \qquad \text{s.t. } L_{h_1^{\rm hom}}^{(N)}(y) = \cdots = L_{h_{m_1}^{\rm hom}}^{(N)}(y) = L_{h_{m_1+1}^{\rm hom}}^{(N)}(y) = 0, \\ \qquad\quad L_{\varphi_i^J}^{(N)}(y) = 0, L_{\varphi_j^{J'}}^{(N)}(y) = 0 \\ \qquad\quad (i = t(J)+1, \ldots, len(J),\ j \in [t(J)],\ J \subseteq [m_2], |J| \le l), \\ \qquad\quad L_{g_\nu^{\rm hom}}^{(N)} \succeq 0, \forall \nu \in \{0,1\}^{m_2+1}, y_0 = 1. \end{array} \right. \tag{3.12}$$

The dual problem of (3.12) is

$$\begin{cases} p_N^{(2)} := \max\limits_{\gamma \in \mathbb{R}^{n+1}} \ \gamma \\ \qquad\qquad \text{s.t. } \tilde{p}(\tilde{x}) - \gamma \in I^{(N)} + P^{(N)}. \end{cases} \tag{3.13}$$

where $I^{(N)}$ is the ideal defined in (3.10) and

$$P^{(N)} = \left\{ \sum_{\nu \in \{0,1\}^{m_2+1}} \sigma_\nu g_\nu^{\text{hom}} \,\middle|\, \begin{array}{l} \deg(\sigma_\nu g_\nu^{\text{hom}}) \le 2N \\ \sigma_\nu\text{'s are SOS} \end{array} \right\}.$$

**Definition 3.4.1.** *For every set $J = \{j_1, \ldots, j_k\} \subseteq [m_2 + 1]$ with $k \le n - m_1$, let*

$$\Theta_J = \{\tilde{x} \in V(h^{\text{hom}}, g_J^{\text{hom}}) \mid \textit{rank } \begin{bmatrix} \nabla_{\tilde{x}} h^{\text{hom}} & \nabla_{\tilde{x}} g_J^{\text{hom}} \end{bmatrix} < m_1 + |J| + 1\}$$

*and*

$$\Theta = \bigcup_{J \subseteq [m_1+1],\ |J| \le n - m_1} \Theta_J.$$

**Assumption 3.4.2.** (i) $m_1 \le n$; (ii) *For any $u \in \widetilde{S}$ in* (3.4)*, at most $n - m_1$ of $g_1^{hom}(u), \ldots, g_{m_2+1}^{hom}(u)$ vanish;* (iii) *The set $\Theta$ is finite.*

By Theorem 2.2.10 and 3.2.4, we have

**Theorem 3.4.3.** *Suppose Assumption 3.4.2 holds. Then $p^* > -\infty$ in* (3.11) *and there exists $N^* \in \mathbb{N}$ such that $p_N^{(1)} = p_N^{(2)} = p^*$ for all $N \ge N^*$. Furthermore, if one of the conditions in Theorem 3.2.4 holds and the minimum $s^*$ of* (3.2) *is achievable, then $p_N^{(1)} = p_N^{(2)} = r^*$ for all $N \ge N^*$.*

**Corollary 3.4.4.** *If $S = \mathbb{R}^n$ in* (3.1) *and $s^*$ is achievable in* (3.6)*, then there exists $N^* \in \mathbb{N}$ such that $p_N^{(1)} = p_N^{(2)} = r^*$ for all $N \ge N^*$ in* (3.12) *and* (3.13)*.*

*Proof.* Since the only constraint is $\tilde{q} - 1 = 0$ and $\tilde{q}$ is homogeneous, regarding $\nabla\tilde{q}$ and $\tilde{x}$ as vectors in $\mathbb{R}^{n+1}$, then $d \cdot \tilde{q} = \nabla\tilde{q}^T \cdot \tilde{x}$ by Euler's Formula. Thus $\nabla(\tilde{q} - 1) = \nabla\tilde{q} = 0$ implies $\tilde{q} = 0$, i.e. $\Theta = \emptyset$. Hence, Assumption 3.4.2 is always true for (3.6). Then by Corollary 3.2.5 and Theorem 3.4.3, the conclusion follows. □

Theorem 3.4.3 and Corollary 3.4.4 show that we can apply Jacobian SDP relaxations to solve (3.2) in finite steps under some generic assumptions on the feasible set. This result is more interesting theoretically than practically since the additional polynomials $\varphi_j$'s are complicated and the preordering instead of the quadratic module is used in (2.17). Generically, Lasserre's SDP relaxation [39] has finite convergence as shown in [52]. Therefore, in practice, we can use Lasserre's SDP relaxation (Algorithm 2.1.3) to solve (3.2).

In the end of this section, we would like to point out that $s^*$ in (3.2) might not be achievable in some cases. If the infimum of a constrained polynomial optimization is an asymptotic value, some approaches are proposed in [15, 79]. Hence, we can use these approaches to solve (3.2). However, the finite convergence for these methods is unknown.

## 3.5 Numerical Experiments

In this section, we present some numerical examples to illustrate the efficiency of our method for solving (3.1). We use the software *GloptiPoly* [20] to solve (3.12) and (3.13).

### 3.5.1 Unconstrained Rational Optimization

In the following, Example 3.5.1 and 3.5.2 are constructed from *Motzkin* polynomial,

$$M(x_1, x_2, x_3) = x_1^4x_2^2 + x_1^2x_2^4 + x_3^6 - 3x_1^2x_2^2x_3^2, \tag{3.14}$$

which is well-known nonnegative on $\mathbb{R}^3$ but not SOS [65].

**Example 3.5.1.** ([54, Example 2.9])

Consider the minimization problem

$$\min_{x_1,x_2\in\mathbb{R}} r(x_1, x_2) := \frac{x_1^4x_2^2 + x_1^2x_2^4 + 1}{x_1^2x_2^2}. \tag{3.15}$$

Taking $x_3 = 1$ in Motzkin polynomial, we have $x_1^4x_2^2 + x_1^2x_2^4 + 1 - 3x_1^2x_2^2 \geq 0$ on $\mathbb{R}^2$. Since $r(1, 1) = 3$, we have $r^* = 3$ and there are four minimizers $(\pm1, \pm1)$.

However, $x_1^4x_2^2 + x_1^2x_2^4 + 1 - r^*x_1^2x_2^2$ is not SOS. To solve this problem, the authors in [54] used the generalized big ball technique. More specifically, it is assumed that one of the minimizers of (3.15) lies in a ball $B(c, \rho)$ and the numerator and denominator of $r(x_1, x_2)$ have no common real roots on $B(c, \rho)$. However, it is not easy in general to determine the radius $\rho$ of this ball. We now solve this problem by using our method without the assumptions in [54].

We first reformulate the problem as the following polynomial optimization problem by homogenization.

$$\begin{cases} \min\limits_{x_0,x_1,x_2\in\mathbb{R}} \tilde{p}(x_0, x_1, x_2) := x_1^4x_2^2 + x_1^2x_2^4 + x_0^6 \\ \quad \text{s.t. } \tilde{q}(x_0, x_1, x_2) := x_1^2x_2^2x_0^2 = 1. \end{cases}$$

By using Jacobian SDP relaxation, we need 3 more equations:

$$\begin{aligned} \varphi_1(x_0, x_1, x_2) &= 4x_1^3x_2^3x_0^2(x_1^2 - x_2^2) = 0 \\ \varphi_2(x_0, x_1, x_2) &= 4x_1x_2^2x_0(2x_1^4x_2^2 + x_1^2x_2^4 - 3x_0^6) = 0 \\ \varphi_3(x_0, x_1, x_2) &= 4x_1^2x_2x_0(x_1^4x_2^2 + 2x_1^2x_2^4 - 3x_0^6) = 0 \end{aligned}$$

By the condition $x_1^2x_2^2x_0^2 = 1$, the above three equations can be simplified as

$$\begin{aligned} \varphi_1(x_0, x_1, x_2) &= x_1^2 - x_2^2 = 0 \\ \varphi_2(x_0, x_1, x_2) &= 2x_1^4x_2^2 + x_1^2x_2^4 - 3x_0^6 = 0 \\ \varphi_3(x_0, x_1, x_2) &= x_1^4x_2^2 + 2x_1^2x_2^4 - 3x_0^6 = 0 \end{aligned}$$

We need to solve the following new problem

$$\begin{cases} \min\limits_{x_0,x_1,x_2\in\mathbb{R}} x_1^4x_2^2 + x_1^2x_2^4 + x_0^6 \\ \quad \text{s.t. } x_1^2x_2^2x_0^2 - 1 = 0, \quad 2x_1^4x_2^2 + x_1^2x_2^4 - 3x_0^6 = 0, \\ \quad\quad x_1^2 - x_2^2 = 0, \quad x_1^4x_2^2 + 2x_1^2x_2^4 - 3x_0^6 = 0. \end{cases}$$

Using GloptiPoly to solve this problem, we get the following results:

- $N = 3$. The optimum is 3, but extracting global optimal solutions fails.
- $N = 4$. We get 8 optimal solutions for $(x_0, x_1, x_2)$: $(\pm1, \pm1, \pm1)$ from which we get all the optimal solutions to original problem: $(\pm1, \pm1)$.

**Example 3.5.2.** ([54, Example 2.10])

Consider the following problem

$$\min_{x_1,x_2\in\mathbb{R}} r(x_1,x_2) := \frac{p(x_1,x_2)}{q(x_1,x_2)} = \frac{x_1^4+x_1^2+x_2^6}{x_1^2x_2^2}. \tag{3.16}$$

Taking $x_2 = 1$ in Motzkin polynomial (3.14), we have $r^* = 3$ with 4 minimizers $(\pm 1, \pm 1)$. The denominator and numerator have real common root $(0, 0)$. In [54], the SDP relaxation extracts 6 solutions, 2 of which are not global minimizers but the common roots of $p(x)$ and $q(x)$. We reformulate it as the following polynomial optimization and solve it by Jacobian SDP relaxation.

$$\begin{cases} \min\limits_{x_0,x_1,x_2\in\mathbb{R}} \tilde{p}(x_0,x_1,x_2) := x_1^4x_0^2 + x_1^2x_0^4 + x_2^6 \\ \quad \text{s.t. } \tilde{q}(x_0,x_1,x_2) := x_1^2x_2^2x_0^2 = 1. \end{cases} \tag{3.17}$$

Using GloptiPoly, we can still extract 8 solutions of (3.17) and obtain all the 4 optimal solutions of (3.16) as in Example 3.5.1. In our method, the constraint $\tilde{q}(x_0, x_1, x_2) = 1$ prevents extracting the common real roots of $p(x)$ and $q(x)$. This example also shows that condition (b) in Theorem 3.2.8 is only sufficient but not necessary. □

The following example is generated from the *Robinson* polynomial

$$x_1^6 + x_2^6 + x_3^6 - (x_1^4x_2^2 + x_1^2x_2^4 + x_1^4x_3^2 + x_1^2x_3^4 + x_2^4x_3^2 + x_2^2x_3^4) + 3x_1^2x_2^2x_3^2$$

which is nonnegative on $\mathbb{R}^3$ but not SOS [65].

**Example 3.5.3.** Consider the following problem

$$\min_{x_1,x_2\in\mathbb{R}} r(x_1,x_2) := \frac{p(x_1,x_2)}{q(x_1,x_2)} = \frac{x_1^6+x_2^6+3x_1^2x_2^2+1}{x_1^2(x_2^4+1)+x_2^2(x_1^4+1)+(x_1^4+x_2^4)}. \tag{3.18}$$

Taking $x_3 = 1$ in Robinson polynomial, we have $p(x_1, x_2) - q(x_1, x_2) \geq 0$ on $\mathbb{R}^2$. Since $r(1, 1) = 1$, $r^* = 1$. We reformulate it as the following polynomial optimization problem:

$$\begin{cases} \min\limits_{x_0,x_1,x_2\in\mathbb{R}} x_1^6 + x_2^6 + 3x_1^2x_2^2x_0^2 + x_0^6 \\ \quad \text{s.t. } x_1^2(x_2^4+x_0^4) + x_2^2(x_1^4+x_0^4) + x_0^2(x_1^4+x_2^4) = 1. \end{cases} \tag{3.19}$$

The numerical results we obtain are:

- For relaxation order $N = 5, 6$, we get the optimum $s^* = 1$, but the minimizers can not be extracted.

- For relaxation order $N = 7$, we extract 20 approximate minimizers of (3.19):

$$(-0.0000, \pm 0.8909, \pm 0.8909), \qquad (\pm 0.8909, \pm 0.8909, -0.0000),$$
$$(\pm 0.8909, -0.0000, \pm 0.8909), \qquad (\pm 0.7418, \pm 0.7418, \pm 0.7418).$$

The above solutions correspond to the exact minimizers of (3.19):

$$\left(0, \pm\frac{1}{\sqrt[6]{2}}, \pm\frac{1}{\sqrt[6]{2}}\right), \qquad \left(\pm\frac{1}{\sqrt[6]{2}}, \pm\frac{1}{\sqrt[6]{2}}, 0\right),$$
$$\left(\pm\frac{1}{\sqrt[6]{2}}, 0, \pm\frac{1}{\sqrt[6]{2}}\right), \qquad \left(\pm\frac{1}{\sqrt[6]{6}}, \pm\frac{1}{\sqrt[6]{6}}, \pm\frac{1}{\sqrt[6]{6}}\right).$$

There are four solutions with the first coordinate $x_0^* = 0$ which indicate that minimum $r^* = 1$ is also an asymptotic value at $\infty$ by Theorem 3.2.8. In fact,

$$\lim_{x_1, x_2 \to \infty} \frac{p(x_1, x_2)}{q(x_1, x_2)} = 1 = r^*.$$

From the other 16 solutions, according to (a) in Theorem 3.2.8, we get 8 global minimizers of (3.18): $(\pm 1, \pm 1)$, $(\pm 1, 0)$, $(0, \pm 1)$. □

**Example 3.5.4.** ([54, Example 3.4])

Suppose function $\psi(z)$ and $\phi(z)$ are monic complex univariate polynomials of degree $m$ such that:

$$\psi(z) = z^m + \psi_{m-1} z^{m-1} + \cdots + \psi_1 z + \psi_0$$

$$\phi(z) = z^m + \phi_{m-1} z^{m-1} + \cdots + \phi_1 z + \phi_0$$

It is shown in [30] that finding nearest GCDs becomes the following global minimization of rational functions

$$\min_{x_1, x_2 \in \mathbb{R}} \frac{p(x_1, x_2)}{q(x_1, x_2)} = \frac{|\psi(x_1 + i x_2)|^2 + |\phi(x_1 + i x_2)|^2}{\sum_{k=0}^{m-1} (x_1^2 + x_2^2)^k} \tag{3.20}$$

where $\deg(p) = 2m$ and $\deg(q) = 2(m-1)$. Let

$$\psi(z) = z^3 + z^2 - 2, \quad \phi(z) = z^3 + 1.5z^2 + 1.5z - 1.25.$$

By using our method, for relaxation order $N = 5$, we get four optimal solutions of the polynomial optimization reformulated from (3.20) by homogenization:

$$(0.7050, -0.7073, \pm 0.7763), \quad (-0.7050, 0.7073, \pm 0.7763).$$

The corresponding minimizers of (3.20) are

$$(x_1 \approx -1.0033, x_2 \approx \pm 1.1011) \quad (x_1 \approx -1.0033, x_2 \approx \pm 1.1011).$$

Hence there are two global minimizers $(-1.0033, \pm 1.1011)$ which are the same as in [54]. The minimum is $r^* \approx 0.0643$. □

### 3.5.2 Constrained Rational Optimization

In this subsection, we give some numerical examples of minimizing of rational functions with polynomial inequality constraints. We first consider an example for which $p(x)$ and $q(x)$ have common roots.

**Example 3.5.5.** ([54]) Consider the following optimization problem

$$\begin{cases} \min\limits_{x\in\mathbb{R}} r(x) := \dfrac{1+x}{(1-x^2)^2} \\ \text{s.t. } (1-x^2)^3 \geq 0. \end{cases} \tag{3.21}$$

As shown in [54], the global minimum $r^* = \frac{27}{32} \approx 0.8438$ and the minimizer $x^* = -\frac{1}{3} \approx -0.3333$. If the denominator and numerator have common roots, SDP relaxation method proposed in [54] can not guarantee to converge to the minimum. Reformulating the above problem by homogenization, we get

$$\begin{cases} \min\limits_{x_0,x_1\in\mathbb{R}} x_0^4 + x_1 x_0^3 \\ \quad \text{s.t. } x_0^4 - 2x_1^2x_0^2 + x_1^4 = 1, \\ \qquad x_0^6 - 3x_0^4x_1^2 + 3x_0^2x_1^4 - x_1^6 \geq 0, \ x_0 \geq 0. \end{cases} \tag{3.22}$$

For relaxation order $N = 7$, by the Jacobian SDP relaxation, we get the optimal solution of (3.22) $\tilde{x}^* \approx (1.0607, -0.3536)$ and the minimum $s^* \approx 0.8437$. According to (a) in Theorem 3.2.8, we find the minimizer of (3.21): $x^* \approx -0.3334$. □

We next consider Example 3.5.3 with some constraints.

**Example 3.5.6.** Consider optimization

$$r^* := \min_{x\in S} \frac{p(x_1,x_2)}{q(x_1,x_2)} = \frac{x_1^6+x_2^6+3x_1^2x_2^2+1}{x_1^2(x_2^4+1)+x_2^2(x_1^4+1)+(x_1^4+x_2^4)}. \tag{3.23}$$

(1) $S = \{(x_1,x_2) \in \mathbb{R}^2 \mid x_1^2+x_2^2 \le 1\}$. It is easy to check that $S$ is closed at $\infty$. By Theorem 3.2.4, $r^*$ is equal to the optimum of the following polynomial optimization problem:

$$\begin{cases} s^* = \min\limits_{x_0,x_1,x_2\in\mathbb{R}} x_1^6+x_2^6+3x_1^2x_2^2x_0^2+x_0^6 \\ \qquad\qquad \text{s.t. } x_1^2(x_2^4+x_0^4)+x_2^2(x_1^4+x_0^4)+x_0^2(x_1^4+x_2^4)=1, \\ \qquad\qquad x_0^2-x_1^2-x_2^2 \ge 0,\ x_0 \ge 0. \end{cases} \tag{3.24}$$

Using Jacobian SDP relaxation, for relaxation order $N=7$, we get $r^* = s^* = 1$ and four approximate minimizers:

$$(0.8909, \pm 0.8909, -0.0000), \qquad (0.8909, -0.0000, \pm 0.8909),$$

which correspond to the exact minimizers:

$$\left(\frac{1}{\sqrt[6]{2}}, \pm\frac{1}{\sqrt[6]{2}}, 0\right), \quad \left(\frac{1}{\sqrt[6]{2}}, 0, \pm\frac{1}{\sqrt[6]{2}}\right).$$

Then we get four minimizers of (3.23): $(\pm 1, 0)$, $(0, \pm 1)$.

(2) $S = B(0,\sqrt{2})^c = \{(x_1,x_2) \in \mathbb{R}^2 \mid x_1^2+x_2^2 \ge 2\}$. $S$ is noncompact but closed at $\infty$. By Theorem 3.2.4, we solve the following equivalent optimization:

$$\begin{cases} s^* := \min\limits_{x_0,x_1,x_2\in\mathbb{R}} x_1^6+x_2^6+3x_1^2x_2^2x_0^2+x_0^6 \\ \qquad\qquad \text{s.t. } x_1^2(x_2^4+x_0^4)+x_2^2(x_1^4+x_0^4)+x_0^2(x_1^4+x_2^4)=1, \\ \qquad\qquad x_1^2-x_2^2-2x_0^2 \ge 0,\ x_0 \ge 0. \end{cases} \tag{3.25}$$

For relaxation order $N=7$, we get the minimum $s^* = r^* = 1$ and 8 approximate minimizers of (3.25):

$$(0.0002, \pm 0.8909, \pm 0.8909), \qquad (0.7418, \pm 0.7419, \pm 0.7419),$$

which correspond to the exact minimizers:

$$\left(0, \pm\frac{1}{\sqrt[6]{2}}, \pm\frac{1}{\sqrt[6]{2}}\right), \quad \left(\frac{1}{\sqrt[6]{6}}, \pm\frac{1}{\sqrt[6]{6}}, \pm\frac{1}{\sqrt[6]{6}}\right).$$

The former solutions indicate that $r^* = 1$ is also an asymptotic values at $\infty$. From the latter solutions, we get four minimizers of (3.23): $(\pm 1, \pm 1)$.

□

Chapter 3, in full, is a reprint of the material as it appears in the article "Minimizing Rational Functions by Exact Jacobian SDP relaxation Applicable to Finite Singularities" by Feng Guo, Li Wang and Guangming Zhou, in the Journal of Global Optimization in volume 58, No.2(2014). The dissertation author was one of the authors of this paper.

# Chapter 4

# Semi-Infinite Polynomial Programming

## 4.1 Introduction

Consider the Semi-Infinite Polynomial Programming (SIPP) problem:

$$(P): \begin{cases} f^* := \min\limits_{x \in X} f(x) \\ \qquad \text{s.t. } g(x,u) \geq 0, \ \forall \ u \in U, \end{cases} \tag{4.1}$$

where

$$X = \{x \in \mathbb{R}^n \mid \theta_1(x) \geq 0, \cdots, \theta_{m_2}(x) \geq 0\},$$
$$U = \{u \in \mathbb{R}^p \mid h_1(u) \geq 0, \cdots, h_{m_1}(u) \geq 0\}.$$

Here $f(x), \theta_i(x)$ are polynomials in $x \in \mathbb{R}^n$, $h_j(u)$ are polynomials in $u \in \mathbb{R}^p$ and $g(x,u)$ is a polynomial in $(x,u) \in \mathbb{R}^n \times \mathbb{R}^p$. Throughout this chapter, we assume that $X$ is compact and $U$ is an infinite index set, i.e., there are infinitely many constraints in $(P)$. The SIPP problem is a special subclass of the *semi-infinite programming* (SIP) which has many applications, e.g., Chebyshev approximation, maneuverability problems, some mathematical physics problems and so on [21, 44]. There are various algorithms for SIP problems based on discretization schemes of $U$, such as central cutting plane method [12], Newton's method [73], SQP methods [75] and the like. Most of algorithms for SIP problems, however, are only locally

convergent or globally convergent under some strong assumptions, like convexity or linearity, few of them are specially designed for SIPP problems exploiting features of polynomial optimization problems. Parpas and Rustem [60] proposed a discretization like method to solve min-max polynomial optimization problems, which can be reformulated as SIPP problems. Using a polynomial approximation and an appropriate hierarchy of semidefinite relaxations, Lasserre presented an algorithm to solve the generalized SIPP problems in [41].

One main difficulty in solving a SIP problem is that there are infinite number of constraints. How to deal with the infinite index set $U$ is the key difference among various SIP algorithms. Exchange method is commonly used in SIP computation, and is regarded as the most efficient method on solving SIP problems [21, 44]. The general steps of exchange method are determined algorithmically as follows [21]. Given a subset $U_k \subseteq U$ in iteration $k$ with $|U_k| < \infty$, compute at least one global solution $x^k$ of

$$\min_{x \in X} \ f(x) \quad \text{s.t. } g(x,u) \geq 0, \ \forall \ u \in U_k, \tag{4.2}$$

and solutions $u_1, \ldots, u_t$ of the subproblem

$$g^k := \min_{u \in U} \ g(x^k, u). \tag{4.3}$$

If $g^k \geq 0$, stop; otherwise, set $U_{k+1} = U_k \cup \{u_1, \ldots, u_t\}$ and go to next iteration. Therefore, to successfully apply exchange method to solve SIPP problems, we need to globally solve subproblems (4.2)-(4.3) and extract global minimizers in each iteration. In next section, we will present the SDP relaxation methods for SIPP problems in detail and discuss the global convergence properties.

## 4.2 SIPP with Compact Index Set

The two SDP relaxation algorithms shown in Sections 2.1 and 2.2 provide strong tools to globally solve polynomial optimization problems with finitely many constraints. In this section, we will discuss how to use them to solve SIPP problems globally.

### 4.2.1 A Semidefinite Relaxation Algorithm

The specific description of exchange method with SDP relaxations for SIPP problems is shown in the following.

**Algorithm 4.2.1.** (Semidefinite Relaxations for SIPP)
***Input:*** *Objective function $f(x)$, constraint function $g(x,u)$, semi-algebraic sets $X$, $U$, tolerance $\epsilon$ and maximum iteration number $k_{\max}$.*
***Output:*** *Global optimum $f^*$ and set $X^*$ of minimizers of problem $(P)$.*

*Step 1 Choose random $u_0 \in U$ and let $U_0 = \{u_0\}$. Set $X^* = \emptyset$ and $k = 0$.*

*Step 2 Use Algorithm 2.1.3 to solve*

$$(P_k): \begin{cases} f_k^{\min} := \min\limits_{x \in X} f(x) \\ \qquad \text{s.t. } g(x,u) \geq 0, \ \forall\ u \in U_k. \end{cases} \tag{4.4}$$

*Let $S_k = \{x_1^k, \cdots, x_{r_k}^k\}$ be the set of the global minimizers of problem $(P_k)$.*

*Step 3 Set $U_{k+1} = U_k$. For $i = 1, \cdots, r_k$,*

(a) *Use Algorithm 2.2.3 to solve*

$$(Q_i^k): \quad g_i^k := \min_{u \in U} g(x_i^k, u). \tag{4.5}$$

*Let $T_i^k = \left\{u_{i,j}^k,\ j = 1, \cdots, t_i^k\right\}$ be the set of global minimizers of $(Q_i^k)$.*

(b) *Update $U_{k+1} = U_{k+1} \bigcup T_i^k$.*

(c) *If $g_i^k \geq -\epsilon$, then update $X^* = X^* \bigcup \{x_i^k\}$.*

*Step 4 If $X^* \neq \emptyset$ or $k > k_{\max}$, stop;*

*otherwise, set $k = k + 1$ and go back to Step 2.*

**Remark 4.2.2.** Subproblems $(P_k)$ and $(Q_i^k)$ in Algorithm 4.2.1 can be solved by both Algorithm 2.1.3 and 2.2.3. Finite convergence can be guaranteed by Algorithm 2.2.3 which, however, produces SDPs of size exponentially depending on the number of the constraints. Since $U_k$ enlarges as $k$ increases, subproblem $(P_k)$

consequently becomes hard to be solved by Algorithm 2.2.3. Therefore, we solve $(P_k)$ by Algorithm 2.1.3 which is also proved to have finite convergence generically [52]. Because the index set $U$ is fixed and compact, Algorithm 2.2.3 is a better choice for solving $(Q_i^k)$.

### 4.2.2 Global Convergence Properties

In this subsection, we discuss the global convergence properties of Algorithm 4.2.1.

**Proposition 4.2.3** (Monotonic Property)**.**
*For optimal values of* $(P_k)$ *in* (4.4)*, we have*

$$f_1^{\min} \leq \cdots \leq f_k^{\min} \leq f_{k+1}^{\min} \leq \cdots \leq f^*. \tag{4.6}$$

*Proof.* Because

$$U_1 \subseteq \cdots \subseteq U_k \subseteq U_{k+1} \subseteq \cdots \subseteq U.$$

So the feasible sets of $(P_k)$ and $(P)$ satisfy

$$K \subseteq \cdots \subseteq K_{k+1} \subseteq K_k \subseteq \cdots \subseteq K_1,$$

we obtain the conclusion. □

We have the following convergence analysis of Algorithm 4.2.1:

**Theorem 4.2.4.** *Suppose that* $X$ *is compact. If at each step* $k$*,*

(a) *subproblems* $(P_k)$ *and each* $(Q_i^k)$ *are globally solved,*

(b) *intermediate results* $S_k$ *and at least one* $T_i^k$ *are nonempty,*

*then either Algorithm 4.2.1 stops with solutions to* $(P)$ *in a finite number of iterations or for any sequence* $\{x^k\}$ *with* $x^k \in S_k$*, there exists at least one limit point as* $k$ *increases and each of them solves* $(P)$*.*

*Proof.* At each step, if (a) holds, then global optima $f_k^{\min}$ and $g_i^k$ are obtained and monotonic property (4.6) is true. Additionally, if (b) is satisfied, then Algorithm 4.2.1 either stops in a finite number of iterations or proceeds without interrupt as $k$ increases.

If Algorithm 4.2.1 stops at $k$-th iteration with $k < k_{\max}$, then $g_i^k \geq 0$ for some $i$, which implies that the associated $x_i^k$ is feasible for $(P)$. Moreover, $x_i^k$ is a global minimizer of $(P)$ by (4.6). Now we assume $g_i^k < 0$ for each $k$ and $i$ which implies $T_i^k \not\subseteq U_k$ and $U_k \subset U_{k+1}$ for all $k$. The following argument is based on the proof of [21, Theorem 7.2]. For any $x \in X$, define

$$v(x) := \min\{g(x,u), u \in U\}.$$

Obviously, $v(x)$ is continuous. Fix a sequence $\{x^k\}$ with $x^k \in S_k$, then a limit point $\bar{x} \in X$ always exists since $X$ is compact. Without loss of generality, assume $x^k \to \bar{x}$. By (4.6), it suffices to prove that $\bar{x}$ is feasible for $(P)$. Let $v(x^k) = g(x^k, u^k)$ and $X_k$ be the feasible set of $(P_k)$. Since $U_k \subset U_{k+1}$, we have $\bar{x} \in \cap_{k=1}^{\infty} X_k$ and therefore $g(\bar{x}, u^k) \geq 0$. Then

$$\begin{aligned} v(\bar{x}) &= v(x^k) + [v(\bar{x}) - v(x^k)] \\ &= g(x^k, u^k) + [v(\bar{x}) - v(x^k)] \\ &\geq [g(x^k, u^k) - g(\bar{x}, u^k)] + [v(\bar{x}) - v(x^k)]. \end{aligned}$$

By the continuity of $v$ and $g$, we have $v(\bar{x}) \geq 0$, i.e., $\bar{x}$ is feasible for $(P)$. □

If $X$ and $U$ are compact, then the optima of $(P_k)$ and $(Q_i^k)$ are achievable. By applying SDP relaxations Algorithm 2.1.3 and Algorithm 2.2.3 to $(P_k)$ and $(Q_i^k)$, (a) and (b) are *generically* satisfied no matter what initial $U_0$ we choose. In section 4.3, we will consider the case when $U$ is noncompact for which the convergence of Algorithm 4.2.1 might fail if we choose an arbitrary initial $U_0$ (Example 4.3.1). We will deal with this issue by the technique of homogenization.

### 4.2.3 Numerical Experiments

This subsection presents some numerical examples to illustrate the efficiency of Algorithm 4.2.1. The computation is implemented with Matlab 7.12 on

a Dell 64-bit Linux Desktop running CentOS (5.6) with 8GB memory and Intel(R) Core(TM) i7 CPU 860 2.8GHz. Algorithm 4.2.1 is implemented with software Gloptipoly [20]. SeDuMi [74] is used as a standard SDP solver. Throughout the computational experiments, we set parameters $k_{\max} = 15$, $\epsilon = 10^{-4}$ in Algorithm 4.2.1. After Algorithm 4.2.1 terminates, let $X^*$ be the output set of global minimizers of $(P)$, $f^*$ be the value of the objective function $f$ over $X^*$ and Iter be the number of iterations Algorithm 4.2.1 has proceeded. Let

$$\mathrm{Obj}_{\mathbf{2}} := \min_{x^* \in X^*} \min_{u \in U} \; g(x^*, u).$$

By the discussion in Subsection 4.2.1, the global minimizers in $X^*$ can be certified by inequality $\mathrm{Obj}_{\mathbf{2}} \geq -\epsilon$.

**(1) Examples of small SIPP problems**

We test some small examples taken from [3, Appendix A]. For nonpolynomial functions, e.g., sine, cosine or exponential function, we use their Taylor polynomial approximations. Let $X = [-100, 100]^n$. Test results are reported in Table 4.1. The Iter column in Table 4.1 indicates that Algorithm 4.2.1 takes a very few steps to find the global minimizer which are certified by the $\mathrm{Obj}_2$ column.

**Table 4.1**: Computational results for small SIPP problems.

| No. | $x^*$ | Iter | $f^*$ | $\mathrm{Obj}_2$ |
|---|---|---|---|---|
| Example 4.2.5 | (-0.0008, 0.4999) | 2 | -0.2504 | 6.4744e-7 |
| Example 4.2.6 | (-0.7500, -0.6180) | 3 | 0.1945 | 3.5305e-7 |
| Example 4.2.7 | (-0.1514, -1.7484, 2.5725) | 2 | 9.6973 | 7.8870e-5 |
| Example 4.2.8 | (-1,0,0) | 2 | 1 | 6.2320e-5 |
| Example 4.2.9 | (0,0) | 2 | 0 | -1.1578e-12 |
| Example 4.2.10 | (0,0,0) | 2 | 4 | -4.7070e-12 |
| Example 4.2.11 | (0,0) | 2 | 0 | 1.9285e-12 |

**Example 4.2.5.** *Let* $U = [0, 2]$ *and*

$$f(x) = \frac{1}{3}x_1^2 + \frac{1}{2}x_1 + x_2^2 - x_2, \quad g(x, u) = -x_1^2 - 2x_1x_2u^2 + \sin(u).$$

*Replace the function* $\sin(u)$ *by* $u - \frac{u^3}{6}$.

**Example 4.2.6.** *Let* $U = [0, 1]$ *and*

$$f(x) = \frac{1}{3}x_1^2 + x_2^2 + \frac{1}{2}x_1, \quad g(x, u) = -(1 - x_1^2u^2)^2 + x_1u^2 + x_2^2 - x_2.$$

**Example 4.2.7.** *Let* $U = [0, 1]$ *and*

$$f(x) = x_1^2 + x_2^2 + x_3^2, \quad g(x, u) = -x_1 - x_2e^{x_3u} - e^{2u} + 2\sin(4u).$$

*Replace function* $e^{x_3u}$ *by* $1 + x_3u + \frac{1}{2}x_3^2u^2 + \frac{1}{6}x_3^3u^3 + \frac{1}{24}x_3^4u^4$, *function* $e^{2u}$ *by* $1 + 2u + 2u^2 + \frac{4}{3}u^3 + \frac{2}{3}u^4$, *and function* $\sin(4u)$ *by* $4u - \frac{32}{3}u^3$.

**Example 4.2.8.** *Let* $U = [0, 1]^2$ *and*

$$f(x) = x_1^2+x_2^2+x_3^2, \quad g(x, u) = -x_1(u_1+u_2^2+1)-x_2(u_1u_2-u_2^2)-x_3(u_1u_2+u_2^2+u_2)-1.$$

**Example 4.2.9.** *Let* $U = [0, \pi]$ *and*

$$f(x) = x_2^2 - 4x_2, \quad g(x, u) = -x_1\cos(u) - x_2\sin(u) + 1.$$

*Replace function* $\sin(u)$ *by* $u - \frac{1}{6}u^3$ *and* $\cos(u)$ *by* $1 - \frac{1}{2}u^2 + \frac{1}{24}u^4$.

**Example 4.2.10.** *Let* $U = [0, \pi]$ *and*

$$\begin{aligned} f(x) &= (x_1 + x_2 - 2)^2 + (x_1 - x_2)^2 + 30\min(0, (x_1 - x_2))^2, \\ g(x, u) &= -x_1\cos(u) - x_2\sin(u) + 1. \end{aligned}$$

*Like in [41], let* $x_3 = \min(0, (x_1-x_2))$, *then* $f(x) = (x_1+x_2-2)^2+(x_1-x_2)^2+30x_3^2$. *We add new constraints* $x_3^2 = (x_1 - x_2)^2$ *and* $x_3 \geq 0$ *in* $X$. *Replace function* $\sin(u)$ *by* $u - \frac{u^3}{6} + \frac{y^5}{5!}$.

**Example 4.2.11.** *Let* $U = [-1, 1]$ *and*

$$f(x) = x_2, \quad g(x, u) = -2x_1^2u^2 + u^4 - x_1^2 + x_2.$$

### (2) Examples of random SIPP problems

We test the performance of Algorithm 4.2.1 on some random SIPP problems which are generated as follows.

**Table 4.2**: Computational results for random SIPP problems

| No. | $n$ | $p$ | $d_1$ | $d_2$ | Inst | $U$ | time (min, max) | | $\text{Obj}_2$ (min, max) | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 3 | 2 | 10 | $U_1$ | 0:00:17 | 0:00:28 | 1.3479 | 2.0779 |
| 2 | 5 | 3 | 2 | 2 | 10 | $U_3$ | 0:00:06 | 0:00:12 | -9.5236e-9 | 0.6343 |
| 3 | 6 | 2 | 2 | 2 | 10 | $U_1$ | 0:00:19 | 0:00:22 | 1.7144 | 2.1185 |
| 4 | 6 | 3 | 2 | 2 | 10 | $U_1$ | 0:00:19 | 0:00:24 | 1.0450 | 1.7220 |
| 5 | 7 | 3 | 3 | 2 | 10 | $U_3$ | 0:00:26 | 0:00:59 | 3.7797e-8 | 0.3198 |
| 6 | 8 | 3 | 2 | 2 | 10 | $U_1$ | 0:04:52 | 0:05:18 | 1.3213 | 1.8438 |
| 7 | 9 | 2 | 2 | 2 | 5 | $U_1$ | 0:45:26 | 0:49:28 | 1.5850 | 2.2807 |
| 8 | 9 | 2 | 2 | 2 | 5 | $U_3$ | 0:44:40 | 0:52:49 | 1.7521e-8 | 2.9119e-7 |
| 9 | 5 | 2 | 2 | 2 | 5 | $U_2$ | 0:57:17 | 1:04:02 | 1.3116e-6 | 1.6986e-5 |

Let $x = (x_1, \ldots, x_n)$ and $u = (u_1, \ldots, u_p)$. Given $d \in \mathbb{N}$, let $[x]_d$ and $[u]_d$ be the vectors of monomials with degree up to $d$ in $\mathbb{R}[x]$ and $\mathbb{R}[u]$, respectively. Denote $\langle [x]_d, [u]_d \rangle$ as the vector obtained by stacking $[x]_d$ and $[u]_d$. Let $f(x) = \eta^T [x]_{2d_1}$ be the objective function where $\eta$ is a Gaussian random vector of matching dimension. Let $g(x,u) = \tau - \langle [x]_{d_2}, [u]_{d_2} \rangle^T M \langle [x]_{d_2}, [u]_{d_2} \rangle$, where $\tau$ is a random number in $[1, 10]$ and $M$ is a random positive semidefinite matrix of matching dimension. Let $X = B_n(0,1)$ be the unit ball in $\mathbb{R}^n$ and $U$ varies among $U_1 = B_p(0,1)$, $U_2 = [-1,1]^p$ and $U_3 = \Delta_p$ where $\Delta_p$ is the $p$ dimensional simplex.

The results using Algorithm 4.2.1 are shown in Table 4.2 where the Inst column denotes the number of randomly generated instances, the consumed computer time is in the format hr:mn:sc with hr (resp. mn, sc) standing for the consumed hours (resp. minutes, seconds). The column $\text{Obj}_2$ shows that Algorithm 4.2.1 successfully solves all the random problems.

### 4.2.4 Application of SIPP to PMI problems

In this subsection, we apply Algorithm 4.2.1 to the following optimization problem with *polynomial matrix inequality* (PMI):

$$f^{\min} := \min_{x \in \mathbb{R}^n} \; f(x) \qquad \text{s.t.} \;\; G(x) \succeq 0, \tag{4.7}$$

where $f(x) \in \mathbb{R}[x]$ and $G(x)$ is an $m \times m$ symmetric matrix with entries $G_{ij}(x) \in \mathbb{R}[x]$. PMI is a special SIPP problem and has been widely arising in control system design, e.g., static output feedback design problems [19]. PMI is also interesting

in optimization theory, e.g., SDP representation of a convex semialgebra set [48]. Some traditional methods for globally solving (4.7) are based on branch-and-bound schemes and alike [14] which, as pointed in [19], are computationally expensive. Recently, some global methods based on SDP relaxations are proposed in [23, 32] as well as in [14] in a dual view.

Define

$$X := \{x \in \mathbb{R}^n \mid G(x) \succeq 0\} \quad \text{and} \quad U := \{u \in \mathbb{R}^m \mid \|u\|^2 = 1\}.$$

Then problem (4.7) is equivalent to the following SIPP problem

$$\begin{cases} \min\limits_{x\in\mathbb{R}^n} f(x) \\ \text{s.t. } g(x,u) = u^T G(x) u \geq 0, \quad \forall\ u \in U. \end{cases} \tag{4.8}$$

Assume the feasible set $X$ is compact, then we can apply Algorithm 4.2.1 to solve SIPP problem (4.8). The following examples show that Algorithm 4.2.1 is efficient to solve PMI problems.

**Example 4.2.12.** Consider the following PMI problem:

$$\begin{cases} \min\limits_{x\in\mathbb{R}^2} f(x) = x_1 + x_2 \\ \text{s.t. } G(x) = \begin{bmatrix} 4 - x_1^2 - x_2^2 & x_1 & x_2 \\ x_1 & x_2^2 - x_1 & x_1 x_2 \\ x_2 & x_1 x_2 & x_1^2 - x_2 \end{bmatrix} \succeq 0. \end{cases} \tag{4.9}$$

The characteristic polynomial of matrix $G(x)$ is:

$$p(t,x) = \det(tI_3 - G(x)) = t^3 - g_1(x)t^2 + g_2(x)t - g_3(x)$$

where

$$\begin{aligned} g_1(x) &= 4 - x_1 - x_2, \\ g_2(x) &= x_1^2 x_2 - 4x_2 - x_1^4 + x_1 x_2 - x_2^4 - 2x_1^2 x_2^2 + x_1 x_2^2 - 4x_1 + 3x_1^2 + 3x_2^2, \\ g_3(x) &= x_1^2 x_2 + 4x_1 x_2 + 2x_1^2 x_2^2 + x_1 x_2^2 - x_1^3 x_2 - 4x_1^3 + x_2^2 x_1^3 - x_2^3 x_1 - 4x_2^3 \\ &\quad - x_1^4 - x_2^4 + x_1^5 + x_2^5 + x_1^2 x_2^3. \end{aligned}$$

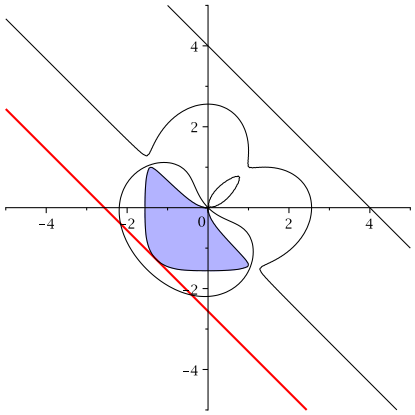


**Figure 4.1**: Feasible region of PMI problem (4.9) in Example 4.2.12.

According to Descartes' rule of signs [19], the feasible set of (4.9) is

$$\left\{x \in \mathbb{R}^2 \mid g_1(x) \geq 0, \quad g_2(x) \geq 0, \quad g_3(x) \geq 0\right\}$$

which is shown shaded in Figure 4.1. We first reformulate (4.9) as a SIPP problem (4.8), then apply Algorithm 4.2.1 to it. After 5 iterations, we get a global minimizer $x^* \approx (-1.2853, -1.2763)$ which is certified by $\text{Obj}_2 = -1.4523 \times 10^{-4}$. The accuracy of this result can be seen from Figure 4.1. □

**Example 4.2.13.** Consider the following PMI problem:

$$\begin{cases} \min\limits_{x \in \mathbb{R}^2} f(x) = x_1 - x_2 \\ \text{s.t. } G(x) = \begin{bmatrix} 10 - x_1^2 - x_2^2 & x_1 & -x_1^2 + x_2 & x_2 + 3 \\ x_1 & x_2^2 & x_1 - x_2^2 & x_1 \\ -x_1^2 + x_2 & x_1 - x_2^2 & x_1 + 2x_2^2 & x_2 \\ x_2 + 3 & x_1 & x_2 & x_2^2 \end{bmatrix} \succeq 0. \end{cases} \tag{4.10}$$

Similar to Example 4.2.12, we obtain the feasible set of (4.10) by Descartes' rule of signs [19] and show it shaded in Figure 4.2. Applying Algorithm 4.2.1 to the reformulation (4.8) of problem (4.10), we get global minimizer $x^* \approx (0.5093, -1.0678)$

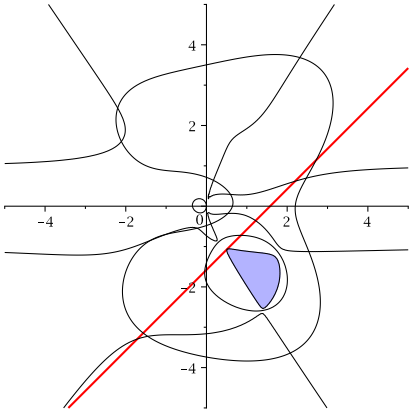


**Figure 4.2**: Feasible region of PMI problem (4.10) in Example 4.2.13.

and minimum $f(x^*) \approx 1.5771$ which are certified by $\text{Obj}_2 = -9.4692 \times 10^{-5}$. From Figure 4.2, we can see this result is accurate. □

We end this section by pointing out a trick hidden in the reformulation (4.8) of (4.7). PMI optimization problem (4.7) can be regarded as a SIPP problem with noncompact index set $\widetilde{U} = \mathbb{R}^m$. Since the constraint function $g(x, u)$ is homogenous in $u$, we can restrict $\widetilde{U}$ to the unit sphere $U$. By Theorem 4.2.4, to guarantee the convergence of Algorithm 4.2.1, the optimum of $(Q_i^k)$ needs to be achievable for each $k$ which might fail if $U$ is noncompact. The reformulation (4.8) of (4.7) gives us a clue for dealing with SIPP with noncompact $U$ by the technique of homogenization. We will go into detail about this technique in next section.

## 4.3 SIPP with Noncompact Index Set

In this section, by homogenization technique, we reformulate the SIPP problem $(P)$ with noncompact index set $U$ as the problem $(\widetilde{P})$ with compact index set $\widetilde{U}$ which can be globally solved by Algorithm 4.2.1. Under the assumption that set $U$ is closed at $\infty$ which is a generic condition, we show the two problems are

equivalent.

### 4.3.1 Motivation

At some $k$-th iteration of Algorithm 4.2.1, if the global minima $g_i^k$ of $(Q_i^k)$ are not achievable for all $x_i^k \in S_k$, then either

case 1. $T_i^k = \emptyset$, then $U_{k+1}$ can not be updated and consequently $S_{k+1}$ remains the same as $S_k$, or

case 2. $U_{k+1}$ is updated by $T_i^k$ which consists of KKT points or singular points of the feasible set of $(Q_i^k)$ rather than global minimizers.

As we have discussed in Subsection 4.2.1, the convergence property of Algorithm 4.2.1 might fail or wrong global minimizers might be outputted if the above cases happen. For example,

**Example 4.3.1.** Consider the following problem:

$$\begin{cases} f^* := \min\limits_{x_1,x_2\in\mathbb{R}} \ -x_1 - x_2 \\ \qquad\quad \text{s.t. } x_1(u_1^2-1) + (x_2 - u_1u_2)^2 \geq 0, \ \forall\ u_1, u_2 \in \mathbb{R}, \\ \qquad\qquad x_1^2 + x_2^2 = 2. \end{cases} \tag{4.11}$$

We choose $u_1, u_2$ such that $x_2 - u_1u_2 = 0$. By letting $u_1$ tend to infinity and 0 respectively, we obtain that $x_1 = 0$ for any feasible point $x$. Therefore, there are only two feasible points $(0, \pm\sqrt{2})$ and the global minimum is $-\sqrt{2}$ with minimizer $(0, \sqrt{2})$.

We claim that Algorithm 4.2.1 *fails* to solve (4.11) if we set initial $U_0 = \{(u_1^0, u_2^0)\}$ such that

$$(u_1^0, u_2^0) \notin \mathcal{U} := \{u \in \mathbb{R}^2 \mid u_1u_2 = \sqrt{2}, u_1^2 < 2\sqrt{2} - 2\}.$$

We prove it in the following. First, we show that for any $(u_1, u_2) \in \mathbb{R}^2$ there always exists $(\bar{x}_1, \bar{x}_2)$ with $\bar{x}_1 > 0, \bar{x}_2 > 0$ such that

$$g(\bar{x}, u) := \bar{x}_1(u_1^2 - 1) + (\bar{x}_2 - u_1u_2)^2 \geq 0, \quad \bar{x}_1^2 + \bar{x}_2^2 = 2.$$

This is true if $g((0,\sqrt{2}),u) > 0$ or $g((\sqrt{2},0),u) > 0$ by the continuity of $g(x,u)$. Now we assume

$$g((0,\sqrt{2}),u) \le 0 \quad \text{and} \quad g((\sqrt{2},0),u) \le 0.$$

From the first inequality, we get $u_1u_2 = \sqrt{2}$. Then by the second inequality, we have $u_1^2 \le 1-\sqrt{2}$ which is a contradiction. Therefore, the following subproblem

$$(P_0): \begin{cases} f_0^{\min} := \min\limits_{x\in\mathbb{R}^2} \ -x_1 - x_2 \\ \qquad\qquad \text{s.t. } x_1^2+x_2^2 = 2,\ g(x,u) \ge 0,\ \forall\ u_1,u_2 \in \mathbb{R}, \end{cases}$$

has global minimizer $S_0 = \{(\tilde{x}_1,\tilde{x}_2)\}$ with $\tilde{x}_1 > 0, \tilde{x}_2 > 0$. Then we solve subproblem

$$(Q^0): \quad g^0 := \min_{u\in\mathbb{R}^2} g(\tilde{x},u) = \tilde{x}_1(u_1^2-1) + (\tilde{x}_2 - u_1u_2)^2. \tag{4.12}$$

Obviously, $g^0 = -\tilde{x}_1$ is not achievable. Applying Jacobian SDP relaxation Algorithm 2.2.3, we obtain $T^0 = \{(0,0)\}$ which consists of the only critical point $(0,0)$ of map $g(x^0,u)$ with critical value $\tilde{x}_2^2 - \tilde{x}_1$. If $\tilde{x}_2^2 - \tilde{x}_1 \ge 0$, then Algorithm 4.2.1 terminates and outputs $X^* = \{(\tilde{x}_1,\tilde{x}_2)\}$ which is a wrong solution. Now we assume $\tilde{x}_2^2 - \tilde{x}_1 < 0$ and continue. By Algorithm 4.2.1, $U_1 = \{(\bar{u}_1,\bar{u}_2),(0,0)\}$. Then we go to the next iteration and solve

$$(P_1): \begin{cases} f_1^{\min} := \min\limits_{x\in\mathbb{R}^2} \ -x_1 - x_2 \\ \qquad\qquad \text{s.t. } x_2^2 - x_1 \ge 0,\ g(x,\bar{u}) \ge 0, \\ \qquad\qquad\quad x_1^2+x_2^2 = 2. \end{cases}$$

Let $K_1$ be the feasible set of $(P_1)$, then

case 1. There exists no $(\bar{x}_1,\bar{x}_2) \in K_1$ with $\bar{x}_1 > 0, \bar{x}_2 > 0$. The global minimizer of $(P_1)$ is $S_1 = \{(0,\sqrt{2})\}$ and $g^1 := \min\limits_{u\in\mathbb{R}^2} g((0,\sqrt{2}),u) \ge 0$. Therefore, the correct global solution of (4.11) is outputted. In this case, by the continuity of $g(x,u)$, we have $g((0,\sqrt{2}),\bar{u}) \le 0$ and $g((1,1),\bar{u}) < 0$. From these two inequalities, we get $(\bar{u}_1,\bar{u}_2) \in \mathcal{U}$.

case 2. There exists $(\bar{x}_1,\bar{x}_2) \in K_1$ with $\bar{x}_1 > 0, \bar{x}_2 > 0$. Then the global minimizer of $(P_1)$ is $S_1 = \{(\hat{x}_1,\hat{x}_2)\}$ with $\hat{x}_1 > 0, \hat{x}_2 > 0$. Similar to $g^0$, $g^1$ is not

achievable and $U_1 = \{(\bar{u}_1, \bar{u}_2), (0,0)\}$ can not be updated. Consequently, the same process will be repeated in the following iterations.

Now we have proved the claim. Since the set $\mathcal{U}$ is a subset of a Zariski closed set of $\mathbb{R}^2$, Algorithm 4.2.1 fails if we choose a generic initial $U_0 = \{(u_1, u_2)\}$. □

Hence, Algorithm 4.2.1 might fail to solve SIPP problem $(P)$ if the optima of subproblems $(Q_i^k)$ can not be reached for all $x_i^k \in S_k$ which might happen when $U$ is noncompact. As we have mentioned at the end of Section 4.2, the reformulation (4.8) of (4.7) sheds light on this issue by the technique of homogenization. In the following, we apply this technique to general SIPP problem $(P)$ with noncompact index set $U$.

### 4.3.2 Equivalent Reformulation by Homogenization

For given polynomial $q(u) \in \mathbb{R}[u] := \mathbb{R}[u_1, \ldots, u_p]$ with degree $d = \deg(q)$, let $\tilde{q}(\tilde{u}) = u_0^d q(u/u_0)$ be the homogenization of $q(u)$ where $\tilde{u} = (u_0, u) \in \mathbb{R}^{p+1}$. Define

$$\tilde{g}(x, \tilde{u}) = u_0^{d_g} g(x, u/u_0) \quad \text{where} \quad d_g = \deg_u g(x,u)$$

and

$$\begin{aligned} U &:= \{u \in \mathbb{R}^p | h_1(u) \geq 0, \cdots, h_{m_1}(u) \geq 0\}, \\ U_0 &:= \{\tilde{u} \in \mathbb{R}^{p+1} | \tilde{h}_1(\tilde{u}) \geq 0, \cdots, \tilde{h}_{m_1}(\tilde{u}) \geq 0, u_0 > 0, \|\tilde{u}\|^2 = 1\}, \\ \widetilde{U} &:= \{\tilde{u} \in \mathbb{R}^{p+1} | \tilde{h}_1(\tilde{u}) \geq 0, \cdots, \tilde{h}_{m_1}(\tilde{u}) \geq 0, u_0 \geq 0, \|\tilde{u}\|^2 = 1\}. \end{aligned}$$

**Proposition 4.3.2.** *$q(u) \geq 0$ on $U$ if and only if $\tilde{q}(\tilde{u}) \geq 0$ on* closure$(U_0)$.

*Proof.* "If" direction. Suppose there exists $v \in U$ such that $q(v) < 0$. For $i \in [m_1]$, we have $h_i(v) \geq 0$. Let $\tilde{v} = \left(\frac{1}{\sqrt{1+\|v\|^2}}, \frac{v}{\sqrt{1+\|v\|^2}}\right)$, then

$$\tilde{h}_i(\tilde{v}) = (1 + \|v\|^2)^{-\frac{\deg(h_i)}{2}} h_i(v) \geq 0, \quad i \in [m_1],$$

which implies $\tilde{v} \in U_0$ and

$$\tilde{q}(\tilde{v}) = (1 + \|v\|^2)^{-\frac{d}{2}} q(v) < 0.$$

It contradicts the assumption that $\tilde{q}(\tilde{v}) \geq 0$ on closure$(U_0)$.

"Only if" direction. Let $\tilde{v} = (v_0, v) \in \mathsf{closure}(U_0)$, then there exists a sequence $\tilde{v}^k = (v_0^k, v^k) \in U_0$ such that $\lim\limits_{k\to\infty}(v_0^k, v^k) = (v_0, v)$ with $v_0^k > 0$ for all $k$. We have

$$h_i(v^k/v_0^k) = (v_0^k)^{-\deg(h_i)}\tilde{h}_i(\tilde{v}^k) \geq 0, \quad i \in [m_1], \quad \text{for all } k.$$

Therefore, the sequence $\{v^k/v_0^k\} \in U$ and $q(v^k/v_0^k) \geq 0$. Since $q$ is continuous,

$$\tilde{q}(\tilde{v}) = \lim_{k\to\infty} \tilde{q}(\tilde{v}^k) = \lim_{k\to\infty} (v_0^k)^d q(v^k/v_0^k) \geq 0,$$

which shows $\tilde{q}(\tilde{v}) \geq 0$ on $\mathsf{closure}(U_0)$. The proof is completed. □

**Corollary 4.3.3.** *A polynomial $q(u) \geq 0$ on $\mathbb{R}^p$ if and only if $\tilde{q}(\tilde{u}) \geq 0$ on $\{\tilde{u} \in \mathbb{R}^{p+1} \mid \|\tilde{u}\|^2 = 1\}$.*

*Proof.* From the proof of Proposition 4.3.2, we can see the inequality $u_0 > 0$ can be removed from $U_0$ such that $q(u) \geq 0$ on $\mathbb{R}^p$ if and only if $\tilde{q}(\tilde{u}) \geq 0$ on

$$\mathsf{closure}(\{\tilde{u} \in \mathbb{R}^{p+1} \mid \|\tilde{u}\|^2 = 1\}) = \{\tilde{u} \in \mathbb{R}^{p+1} \mid \|\tilde{u}\|^2 = 1\}.$$

□

By Proposition 4.3.2, we have the following equivalent reformulation of problem $(P)$:

$$(P_0): \begin{cases} f^* := \min\limits_{x\in X} f(x) \\ \qquad \text{s.t. } \tilde{g}(x, \tilde{u}) \geq 0, \ \forall\ \tilde{u} \in \mathsf{closure}(U_0). \end{cases}$$

Some natural questions arise: how to get the explicit expression of semi-algebraic set $\mathsf{closure}(U_0)$? Is it true that $\mathsf{closure}(U_0) = \widetilde{U}$? Clearly, we have

$$\mathsf{closure}(U_0) \subseteq \widetilde{U}. \tag{4.13}$$

Unfortunately, the equality does not always hold even if set $U$ is compact (cf. [49, Example 5.2]).

**Definition 4.3.4.** ([49]) *$U$ is closed at $\infty$ if $\mathsf{closure}(U_0) = \widetilde{U}$.*

Since it might be hard to express $\mathsf{closure}(U_0)$ for a given particular SIPP problem, we consider to solve the following problem in general:

$$(\widetilde{P}): \begin{cases} \tilde{f}^* := \min\limits_{x\in X} f(x) \\ \qquad \text{s.t. } \tilde{g}(x,\tilde{u}) \geq 0, \ \forall\ \tilde{u} \in \widetilde{U}. \end{cases}$$

As set $\widetilde{U}$ is compact, the semidefinite relaxation Algorithm 4.2.1 in Section 4.2 can successfully solve this problem with any arbitrary initial $U_0$. Next we investigate the relation between problem $(P)$ and problem $(\widetilde{P})$.

We define

$$M = \{x \in \mathbb{R}^n | g(x,u) \geq 0, \ \forall\ u \in U\}.$$
$$\widetilde{M} = \{x \in \mathbb{R}^n | \tilde{g}(x,\tilde{u}) \geq 0, \ \forall\ \tilde{u} \in \widetilde{U}\}.$$

**Proposition 4.3.5.** *We have $\widetilde{M} \subseteq M$ and the equality holds if $U$ is closed at $\infty$.*

*Proof.* By Proposition 4.3.2, we have

$$M = \{x \in \mathbb{R}^n | \tilde{g}(x,\tilde{u}) \geq 0, \ \forall\ \tilde{u} \in \mathsf{closure}(U_0)\}.$$

Then the conclusion follows due to the relationship (4.13). □

Consequently, we have

**Theorem 4.3.6.** *$\tilde{f}^* \geq f^*$ and the equality holds if $U$ is closed at $\infty$.*

Corollary 4.3.3 shows that $U = \mathbb{R}^p$ is closed at $\infty$ and therefore,

**Corollary 4.3.7.** *The following two problems are equivalent:*

$$\begin{cases} \min\limits_{x\in X} f(x) \\ \text{s.t. } g(x,u) \geq 0, \ \forall\ u \in \mathbb{R}^p, \end{cases} \qquad \begin{cases} \min\limits_{x\in X} f(x) \\ \text{s.t. } \tilde{g}(x,\tilde{u}) \geq 0, \ \forall\ \tilde{u} \in \widetilde{U}, \end{cases}$$

*where $\widetilde{U} = \{\tilde{u} \in \mathbb{R}^{p+1} \mid \|\tilde{u}\|^2 = 1\}$.*

**Example 4.3.1** (Continued)**.** We reformulate the problem (4.11) as

$$\begin{cases} \tilde{f}^* := \min\limits_{x_1,x_2\in\mathbb{R}} \ -x_1 - x_2 \\ \qquad \text{s.t. } x_1(u_1^2 - u_0^2) + (x_2u_0^2 - u_1u_2)^2 \geq 0, \ \forall\ \tilde{u} \in \widetilde{U}, \\ \qquad\quad x_1^2 + x_2^2 = 2, \end{cases} \tag{4.14}$$

where $\widetilde{U} = \{(u_0, u_1, u_2) \in \mathbb{R}^3 \mid u_0^2 + u_1^2 + u_2^2 = 1\}$. By choosing $u_0 = 1$, we know $\widetilde{M} \supseteq \{(0, \pm\sqrt{2})\}$ which, obviously, are feasible to (4.14). Therefore, $\tilde{f}^* = f^* = -\sqrt{2}$ with minimizer $(0, \sqrt{2})$. Choosing $U_0 = \{(1, 0, 0)\}$ in Algorithm 4.2.1, Figure 4.3 shows the feasible regions of subproblems $(P_k)$ for iterations $k = 0, 1, \cdots, 5$. Let $h(x) = x_1^2 + x_2^2 - 2$. At $i$-th iteration, the feasible region is defined by

$$K_i := \{x \in \mathbb{R}^2 \mid h(x) = 0, g_0(x) \geq 0, \ldots, g_i(x) \geq 0\}$$

where

$$\begin{aligned}
g_0 &= -x_1 + x_2^2, \\
g_1 &\approx 0.026046 - 0.319630x_1 - 0.19679x_2 + 0.371710x_2^2, \\
g_2 &\approx 0.054893 - 0.115770x_1 - 0.18811x_2 + 0.161160x_2^2, \\
g_3 &\approx 0.068650 - 0.049084x_1 - 0.14992x_2 + 0.081854x_2^2, \\
g_4 &\approx 0.072498 - 0.025711x_1 - 0.12039x_2 + 0.049977x_2^2, \\
g_5 &\approx 0.073368 - 0.018151x_1 - 0.10683x_2 + 0.038891x_2^2.
\end{aligned}$$

For each $i$, the feasible region $K_i$ is the intersection of the left parts of the circle $x_1^2 + x_2^2 = 2$ devided by hyperbolas $g_i(x) = 0, i = 0, \cdots, 5$. From Figure 4.3, we can see the minimizers of subproblems $(P_k)$ converge to $(0, \sqrt{2})$ which is the minimizer of problem (4.11). □

We would like to point out that if $U$ is not closed at $\infty$, we might have $\tilde{f}^* > f^*$. For example,

**Example 4.3.8.** Consider the following SIPP problem:

$$\begin{cases} f^* := \min\limits_{x \in \mathbb{R}} x^2 \\ \qquad \text{s.t. } x(u_1 - u_2 + 1) \geq 0, \ \forall\ u \in U, \\ \qquad\quad x \in [1, 2], \end{cases} \tag{4.15}$$

where

$$U = \{u \in \mathbb{R}^2 : \ u_1^2(u_1 - u_2) - 1 = 0\}.$$

Since for all $u \in U$,

$$g(1, u) = u_1 - u_2 + 1 = \frac{1}{u_1^2} + 1 > 0,$$

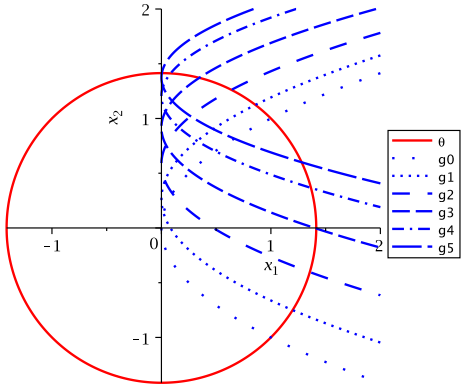


**Figure 4.3**: Feasible region of Example 4.3.1 at each iteration.

$x^* = 1$ is feasible and furthermore the minimizer of problem (4.15). Hence, $f^* = 1$. By definition,

$$\widetilde{U} = \{\tilde{u} \in \mathbb{R}^3 : \ u_1^2(u_1 - u_2) - u_0^3 = 0, \ u_0 \geq 0, \ u_0^2 + u_1^2 + u_2^2 = 1\}.$$

As is shown in [16, 49], $U$ is not closed at $\infty$ because there exists a point $(0, 0, 1) \in \widetilde{U}$ but $(0, 0, 1) \notin \mathsf{closure}(U_0)$. Since for any $x \in [1, 2]$,

$$\tilde{g}(x, (0, 0, 1)) = -x < 0,$$

we have $\widetilde{M} = \emptyset$. Therefore, $\tilde{f}^* = \infty > f^*$. □

Example 4.3.8 shows that the problem $(\widetilde{P})$ might not be equivalent to $(P)$ when set $U$ is not closed at $\infty$. In the following, however, we show that $U$ is closed at $\infty$ in general. In other words, $U$ is closed at $\infty$ if it is defined by generic polynomials.

### 4.3.3 On the Generality of Closedness at $\infty$

Suppose that $U$ is not closed at $\infty$, then by definition there exists $(0, \bar{u}) \in \widetilde{U} \backslash \mathsf{closure}(U_0)$ with $0 \neq \bar{u} \in \mathbb{R}^p$. Let $\hat{h}_i$ denote the homogeneous part of highest

degree of $h_i$ for $i \in [m_1]$ and

$$\{j_1, \ldots, j_\ell\} := \{j \in [m_1] \mid \tilde{h}_j(0, \bar{u}) = \hat{h}_j(\bar{u}) = 0\}.$$

Then $\bar{u}$ is a solution to the polynomial system

$$\hat{h}_{j_1}(\bar{u}) = \cdots = \hat{h}_{j_\ell}(\bar{u}) = \|\bar{u}\|^2 - 1 = 0. \tag{4.16}$$

The Jacobian matrix of the system (4.16) at $\bar{u}$ is

$$A(u) := \begin{bmatrix} \frac{\partial \hat{h}_{j_1}}{\partial u_1}(\bar{u}) & \cdots & \frac{\partial \hat{h}_{j_1}}{\partial u_p}(\bar{u}) \\ \vdots & \vdots & \vdots \\ \frac{\partial \hat{h}_{j_\ell}}{\partial u_1}(\bar{u}) & \cdots & \frac{\partial \hat{h}_{j_\ell}}{\partial u_p}(\bar{u}) \\ 2\bar{u}_1 & \cdots & 2\bar{u}_p \end{bmatrix}$$

By Lemma 3.3.2, we have

**Lemma 4.3.9.** *Suppose $U$ is not closed at $\infty$ and $\ell < p$, then rank $A(u) < \ell + 1$.*

Let $\hat{h}_{m_1+1} := \|\tilde{u}\|^2 - 1$ and $J(\bar{u}) = \{j_1, \ldots, j_\ell, m_1 + 1\}$.

In Section 3.3, we review the definitions and properties of resultants and discriminants for homogenous polynomials. Given inhomogeneous polynomial $h(x) \in \mathbb{R}[x]$, let $\tilde{h}$ denote the homogenization of $h$, i.e., $\tilde{h} = \tilde{h}(\tilde{x}) = x_0^{\deg(h)} h(x/x_0)$. For inhomogeneous polynomials $f_0, f_1, \ldots, f_n \in \mathbb{R}[x]$, the resultant $Res(f_0, f_1, \ldots, f_n)$ is defined to be

$$Res(\tilde{f}_0, \tilde{f}_1, \ldots, \tilde{f}_n).$$

For inhomogeneous polynomials $f_1, \ldots, f_m \in \mathbb{R}[x]$ with $m \leq n$, the discriminant $\Delta(f_1, \ldots, f_m)$ is defined as

$$\Delta(\tilde{f}_1, \ldots, \tilde{f}_m).$$

Then, we have

**Proposition 4.3.10.** *Let $f_0, f_1, \ldots, f_n \in \mathbb{R}[x]$ be inhomogeneous polynomials. Suppose the polynomial system*

$$f_0(x) = f_1(x) = \cdots = f_n(x) = 0$$

*has a solution in $\mathbb{C}^n$, then*

$$Res(f_0, f_1, \ldots, f_n) = 0.$$

*Proof.* If the polynomial system

$$f_0(x) = f_1(x) = \cdots = f_n(x) = 0$$

has a solution $u \in \mathbb{C}^n$, then the polynomial system

$$\tilde{f}_0(\tilde{x}) = \tilde{f}_1(\tilde{x}) = \cdots = \tilde{f}_n(\tilde{x}) = 0$$

has a nonzero solution $(1, u) \in \mathbb{C}^{n+1}$. The conclusion follows by the properties of resultant for homogeneous polynomials . □

**Proposition 4.3.11.** *Let $m \leq n$. The polynomial system*

$$f_1(x) = \cdots = f_m(x) = 0$$

*has a solution $u \in \mathbb{C}^n$ such that the Jacobian matrix of $f_1, \ldots, f_m$ is rank deficient at $u$ if and only if the polynomial system*

$$\tilde{f}_1(\tilde{x}) = \cdots = \tilde{f}_m(\tilde{x}) = 0$$

*has a solution $(1, u) \in \mathbb{C}^{n+1}$ such that the Jacobian matrix of $\tilde{f}_1, \ldots, \tilde{f}_m$ is rank deficient at $(1, u)$.*

*Proof.* Let $d_i = \deg_x(f_i)$, $f_{i,j}$ denote the homogenous part of degree $j$ of polynomial $f_i$ and $\tilde{f}_{i,j} = x_0^{d_i - j} f_{i,j}$ for $i = 1, \cdots, m$ and $j = 0, \cdots, d_i$. Denote

$$\nabla_x := \left\{ \frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_n} \right\} \quad \text{and} \quad \nabla_{\tilde{x}} := \left\{ \frac{\partial}{\partial x_0}, \frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_n} \right\}.$$

The "if" direction is implied by

$$\frac{\partial \tilde{f}_i}{\partial x_j}(1, u) = \frac{\partial f_i}{\partial x_j}(u), \quad i = 1, \cdots, m, \ j = 1, \cdots, n. \tag{4.17}$$

Next we prove the "only if" direction. By assumption, there exists a set of $n$ scalars $c_1, \ldots, c_n$, not all zero, such that

$$\sum_{i=1}^{m} c_i (\nabla_x f_i)(u) = 0$$

which means

$$\sum_{i=1}^{m} c_i \left( \sum_{j=1}^{d_i} \frac{\partial f_{i,j}}{\partial x_k}(u) \right) = 0, \quad k = 1, \cdots, n.$$

Then by Euler's Homogeneous Function Theorem, we have

$$\begin{aligned}
0 &= \sum_{k=1}^{n}\sum_{i=1}^{m} c_i \left( \sum_{j=1}^{d_i} \frac{\partial f_{i,j}}{\partial x_k}(u) u_k \right) \\
&= \sum_{i=1}^{m} c_i \left( \sum_{j=1}^{d_i}\sum_{k=1}^{n} \frac{\partial f_{i,j}}{\partial x_k}(u) u_k \right) \\
&= \sum_{i=1}^{m} c_i \left( \sum_{j=1}^{d_i} j f_{i,j}(u) \right) \\
&= \sum_{i=1}^{m} c_i \left( \sum_{j=1}^{d_i} j f_{i,j}(u) + \sum_{j=0}^{d_i} (d_i - j) f_{i,j}(u) - \sum_{j=0}^{d_i} (d_i - j) f_{i,j}(u) \right) \\
&= \sum_{i=1}^{m} c_i \left( d_i \sum_{j=0}^{d_i} f_{i,j}(u) - \sum_{j=0}^{d_i} \frac{\partial \tilde{f}_{i,j}}{x_0}(1,u) \right) \\
&= \sum_{i=1}^{m} c_i \left( d_i f_i(u) - \frac{\partial \tilde{f}_i}{\partial x_0}(1,u) \right) \\
&= -\sum_{i=1}^{m} c_i \frac{\partial \tilde{f}_i}{\partial x_0}(1,u).
\end{aligned}$$

By combining (4.17), we obtain

$$\sum_{i=1}^{m} c_i (\nabla_{\tilde{x}} \tilde{f}_i)(1,u) = 0$$

which concludes the proof. □

By Proposition 4.3.11 and the properties of discriminant for homogeneous polynomials, we have

**Proposition 4.3.12.** *Let $m \leq n$ and $f_1, \ldots, f_m \in \mathbb{R}[x]$ be inhomogeneous polynomials. Suppose that the polynomial system*

$$f_1(x) = \cdots = f_m(x) = 0$$

*has a solution in $\mathbb{C}^n$ at which the Jacobian matrix of $f_1, \ldots, f_m$ is rank deficient, then*

$$\Delta(f_1, \ldots, f_m) = 0.$$

**Remark 4.3.13.** The reverses of Proposition 4.3.10 and Proposition 4.3.12 are not necessarily true.

By Proposition 4.3.10 and Proposition 4.3.12, we have

**Theorem 4.3.14.** *If $U$ is not closed at $\infty$, then*

(a) *If $|J(\bar{u})| > p$, then for every subset $\{j_1, \cdots, j_{p+1}\} \subseteq J(\bar{u})$,*

$$Res(\hat{h}_{j_1}, \cdots, \hat{h}_{j_{p+1}}) = 0.$$

(b) *If $|J(\bar{u})| \leq p$, then $\Delta(\hat{h}_{j_1}, \cdots, \hat{h}_{j_\ell}, \hat{h}_{m_1+1}) = 0$.*

The above theorem shows that if $U$ is defined by some generic polynomials, then it is closed at $\infty$. Hence, the assumption that $U$ is closed at $\infty$ is a generic condition. Therefore, SIPP problems $(P)$ and $(\widetilde{P})$ are equivalent in general.

### 4.3.4 Numerical Experiment

In this subsection, we present one example to show the efficiency of homogenization method on SIPP with noncompact infinite index set $U$.

**Example 4.3.15.** Consider the following problem

$$\begin{cases} \min\limits_{x \in X} f(x) = x_1^2 + x_2^2 \\ \text{s.t. } g(x,u) = x_1 u_1 + u_2 + x_2 \geq 0, \quad \forall\ u \in U, \end{cases} \tag{4.18}$$

where

$$X := \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 \leq 4\} \text{ and } U := \{(u_1, u_2) \in \mathbb{R}^2 \mid u_1^3 + u_2^3 - 3u_1u_2 \geq 0\}.$$

The set $U$ is shown shaded in Figure 4.4. Since $u_1 + u_2 + 1 = 0$ is the asymptote of the curve $u_1^3 + u_2^3 - 3u_1u_2 = 0$, the inequality $g(x,u) \geq 0$ for all $u \in U$ requires $x_1 = 1$ and $x_2 \geq 1$. Therefore, the feasible set of (4.18) is $\{x \in \mathbb{R}^2 \mid x_1 = 1, 1 \leq x_2 \leq \sqrt{3}\}$ and the global minimizer is $x^* = (1,1)$. It is easy to see that for a given $(\bar{x}_1, \bar{x}_2) \in X$, the global minimum of $g(\bar{x}, u)$ over $U$ is either $-\infty$ or finite but not achievable. Therefore, by the discussion at the beginning of this section, Algorithm 4.2.1 might

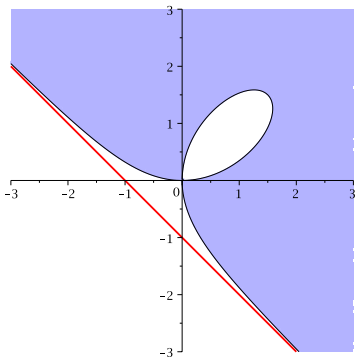


**Figure 4.4**: The feasible region $U$ in Example 4.3.15.

fail to solve (4.18). For example, if we set $U_0 = \{(1,-1)\}$, then we get minimizer $X^* = \{(0.5000, 0.4999)\}$; if $U_0 = \{(1,0)\}$, then $X^* = \{(0.0262, 0.3086) \times 10^{-5}\}$.

Now we use the homogenization technique to reformulate (4.18). First, we show that $U$ is closed at $\infty$. Let

$$U_0 = \{(u_0,u_1,u_2) \in \mathbb{R}^3 | u_1^3 + u_2^3 - 3u_1u_2u_0 \geq 0,\ u_0^2 + u_1^2 + u_2^2 = 1,\ u_0 > 0\},$$
$$\widetilde{U} = \{(u_0,u_1,u_2) \in \mathbb{R}^3 | u_1^3 + u_2^3 - 3u_1u_2u_0 \geq 0,\ u_0^2 + u_1^2 + u_2^2 = 1,\ u_0 \geq 0\}.$$

By definition, if $U$ is not closed at $\infty$, then there exists $(0,\bar{u}_1,\bar{u}_2) \in \widetilde{U} \backslash \mathsf{closure}(U_0)$ which implies

$$\bar{u}_1^3 + \bar{u}_2^3 = 0, \quad \bar{u}_1^2 + \bar{u}_2^2 = 1.$$

Therefore

$$(\bar{u}_1,\bar{u}_2) \in \left\{ \left( -\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right),\ \left( \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2} \right) \right\}.$$

Let

$$\tilde{u}_k := \left( \sqrt{2\varepsilon_k}, -\sqrt{\frac{1}{2} - \varepsilon_k}, \sqrt{\frac{1}{2} - \varepsilon_k} \right), \quad \hat{u}_k := \left( \sqrt{2\varepsilon_k}, \sqrt{\frac{1}{2} - \varepsilon_k}, -\sqrt{\frac{1}{2} - \varepsilon_k} \right).$$

Let $\varepsilon_k \to 0$, then $\tilde{u}_k, \hat{u}_k \in U_0$ for all $k$ large enough and

$$\lim_{k\to\infty} \tilde{u}_k = \left( 0, -\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right), \quad \lim_{k\to\infty} \hat{u}_k = \left( 0, \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2} \right).$$

This shows $U$ is closed at $\infty$. Therefore, by homogenization, we reformulate (4.18) as the following equivalent problem

$$\begin{cases} \min\limits_{x\in X} x_1^2 + x_2^2 \\ \text{s.t. } \tilde{g}(x,\tilde{u}) = x_1u_1 + u_2 + x_2u_0 \geq 0,\ \tilde{u} \in \widetilde{U}. \end{cases}$$

By Algorithm 4.2.1, we find a global minimizer

$$x^* \approx (0.9999, 0.9998) \quad \text{with} \quad \text{Obj}_2 = -9.8148 \times 10^{-7},$$

after several iterations. □

Chapter 4, in full, is a reprint of the material as it appears in the article "Semidefinite Relaxations for Semi-Infinite Polynomial Programming" by Li Wang and Feng Guo, in Computational Optimization and Applications, Volume 58, No.1(2014). The dissertation author was the first author of this paper.

# Chapter 5

# Best Rank-1 Tensor Approximations

## 5.1 Introduction

Let $m$ and $n_1, \ldots, n_m$ be positive integers. A tensor of order $m$ and dimension $(n_1, \ldots, n_m)$ is an array $\mathcal{F}$ that is indexed by integer tuples $(i_1, \ldots, i_m)$ with $1 \leq i_j \leq n_j$ for $j = 1, \ldots, m$, i.e.,

$$\mathcal{F} = (\mathcal{F}_{i_1,\ldots,i_m})_{1 \leq i_1 \leq n_1, \ldots, 1 \leq i_m \leq n_m}.$$

The space of all such tensors with complex (resp., real) entries is denoted as $\mathbb{C}^{n_1 \times \cdots \times n_m}$ (resp., $\mathbb{R}^{n_1 \times \cdots \times n_m}$). Tensors of order $m$ are often called $m$-tensors. When $m$ equals 1 or 2, they are regular vectors or matrices. When $m = 3$ (resp., 4), they are called cubic (resp., quartic) tensors. A tensor $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$ is symmetric if $n_1 = \cdots = n_m$ and

$$\mathcal{F}_{i_1,\ldots,i_m} = \mathcal{F}_{j_1,\ldots,j_m} \quad \text{if} \quad (i_1, \ldots, i_m) \sim (j_1, \ldots, j_m),$$

where $\sim$ means that $(i_1, \ldots, i_m)$ is a permutation of $(j_1, \ldots, j_m)$. We define the norm of a tensor $\mathcal{F}$ as:

$$\|\mathcal{F}\| = \left( \sum_{i_1=1}^{n_1} \cdots \sum_{i_m=1}^{n_m} |\mathcal{F}_{i_1,\ldots,i_m}|^2 \right)^{1/2}. \tag{5.1}$$

If $m = 1$, $\|\mathcal{F}\|$ is vector 2-norm, and if $m = 2$, $\|\mathcal{F}\|$ is matrix Frobenius norm.

Every tensor can be expressed as a linear combination of outer products of vectors. For given vectors $u_1 \in \mathbb{C}^{n_1}, \ldots, u_m \in \mathbb{C}^{n_m}$, their outer product $u_1 \otimes \cdots \otimes u_m$ is the tensor in $\mathbb{C}^{n_1 \times \cdots \times n_m}$ such that for all $1 \le i_j \le n_j$ $(j = 1, \ldots, m)$

$$(u_1 \otimes \cdots \otimes u_m)_{i_1,\ldots,i_m} = (u_1)_{i_1} \cdots (u_m)_{i_m}.$$

For every $\mathcal{F}$ of order $m$, there exist tuples $(u^{i,1}, \ldots, u^{i,m})$ $(i = 1, \ldots, r)$, with each $u^{i,j} \in \mathbb{C}^{n_j}$, $j = 1, \ldots, m$, such that

$$\mathcal{F} = \sum_{i=1}^{r} u^{i,1} \otimes \cdots \otimes u^{i,m}. \tag{5.2}$$

The smallest $r$ in the above is called the rank of $\mathcal{F}$ and is denoted as $\operatorname{rank} \mathcal{F}$. When $\operatorname{rank} \mathcal{F} = r$, (5.2) is called a rank decomposition of $\mathcal{F}$, and we say that $\mathcal{F}$ is a rank-$r$ tensor.

Tensor problems have wide applications in chemometrics, signal processing and high order statistics [8]. For the theory and applications of tensors, we refer to Kolda and Bader's survey [33] and Landsberg's book [38]. When $m \ge 3$, determining ranks of tensors and computing rank decompositions are NP-hard (cf. Hillar and Lim [22]). We refer to Brachat et al. [25], Bernardi et al. [1], Oeding and Ottaviani [57] for tensor decompositions. When $r > 1$, the problem of finding the best rank-$r$ approximation may be ill-posed, because a best rank-$r$ approximation might not exist, as discovered by De Silva and Lim [72]. However, a best rank-1 approximation always exists. It is also NP-hard to compute best rank-1 approximations. This chapter studies best real rank-1 approximation for real tensors. In the following, we will review some existing work for symmetric and nonsymmetric tensors separately.

### 5.1.1 Nonsymmetric Tensors

Given a tensor $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$, we say that a tensor $\mathcal{B}$ is the best rank-1 approximation of $\mathcal{F}$ if it is a minimizer of the least squares problem

$$\min_{\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_m}, \operatorname{rank}(\mathcal{X}) = 1} \|\mathcal{F} - \mathcal{X}\|^2. \tag{5.3}$$

This is equivalent to a homogeneous polynomial optimization problem. For convenience of description, we define the homogeneous polynomial

$$F(x^1,\dots,x^m)=\sum_{1\le i_1\le n_1,\dots,1\le i_m\le n_m}\mathcal{F}_{i_1,\dots,i_m}(x^1)_{i_1}\cdots(x^m)_{i_m}, \tag{5.4}$$

which is in $x^1\in\mathbb{R}^{n_1},\dots,x^m\in\mathbb{R}^{n_m}$. Note that $F(x^1,\dots,x^m)$ is a multi-linear form (a form is a homogeneous polynomial), since it is linear in each $x^j$. De Lathauwer, De Moor and Vandewalle [37] proved the following result.

**Theorem 5.1.1.** ([37, Theorem 3.1]) *For a tensor $\mathcal{F}\in\mathbb{R}^{n_1\times\cdots\times n_m}$, the rank-1 approximation problem (5.3) is equivalent to the maximization problem*

$$\begin{cases}\max\limits_{x^1\in\mathbb{R}^{n_1},\dots,x^m\in\mathbb{R}^{n_m}} & |F(x^1,\dots,x^m)|\\ \text{s.t.} & \|x^1\|=\cdots=\|x^m\|=1,\end{cases} \tag{5.5}$$

*that is, a rank-1 tensor $\lambda\cdot(u^1\otimes\cdots\otimes u^m)$ with $\lambda\in\mathbb{R}$ and each $\|u^i\|=1$, is a best rank-1 approximation for $\mathcal{F}$ if and only if $(u^1,\cdots,u^m)$ is a global maximizer of (5.5) and $\lambda=F(u^1,\dots,u^m)$. Moreover, it also holds that*

$$\|\mathcal{F}-\lambda\cdot(u^1\otimes\cdots\otimes u^m)\|^2=\|\mathcal{F}\|^2-\lambda^2.$$

By Theorem 5.1.1, to find a best rank-1 approximation, it is enough to find a global maximizer of the multi-linear optimization problem (5.5). There exist methods on finding rank-1 approximation, like the alternating least squares (ALS), truncated high order singular value decomposition (T-HOSVD), higher-order power method (HOPM), and Quasi-Newton methods. We refer to Zhang and Golub [82], De Lathauwer, De Moor and Vandewalle [36, 37], Savas and Lim [69] and the references therein. An advantage of these methods is that they can be easily implemented. These kinds of methods typically generate a sequence that converges to a locally optimal rank-1 approximation or even just a stationary point. Even for the lucky cases that they get globally optimal rank-1 approximations, it is usually very difficult to verify the global optimality by these methods.

### 5.1.2 Symmetric Tensors

Let $\mathtt{S}^m(\mathbb{R}^n)$ be the space of real symmetric tensors of order $m$ and in dimension $n$. For a given $\mathcal{F}\in\mathtt{S}^m(\mathbb{R}^n)$, we say that $\mathcal{B}$ is a best rank-1 approximation

of $\mathcal{F}$ if it is a minimizer of the optimization problem

$$\min_{\mathcal{X}\in\mathbb{R}^{n\times\cdots\times n},\operatorname{rank}\mathcal{X}=1} \|\mathcal{F}-\mathcal{X}\|^2. \tag{5.6}$$

When $\mathcal{F}$ is symmetric, Zhang, Ling and Qi [80] showed that (5.6) has a global minimizer that belongs to $\mathbb{S}^m(\mathbb{R}^n)$, i.e., (5.6) has an optimizer that is a symmetric tensor. It is possible that a best rank-1 approximation of a symmetric tensor might not be symmetric. But there is always at least one global minimizer of (5.6) that is a symmetric rank-1 tensor. Therefore, for convenience, by best rank-1 approximation for symmetric tensors, we mean best symmetric rank-1 approximation.

A symmetric tensor in $\mathbb{S}^m(\mathbb{R}^n)$ is rank-1 if and only if it equals $\lambda\cdot(u\otimes\cdots\otimes u)$ for some $\lambda\in\mathbb{R}$ and $u\in\mathbb{R}^n$. For convenience, denote $u^{\otimes m}:=u\otimes\cdots\otimes u$ ($u$ is repeated $m$ times). In the spirit of Theorem 5.1.1 and the work [80], (5.6) is equivalent to the optimization problem

$$\max_{x\in\mathbb{R}^n} \quad |f(x)| \quad \text{s.t.} \quad \|x\|=1, \tag{5.7}$$

where $f(x):=F(x,\cdots,x)$. Therefore, if $u$ is a global maximizer of (5.7) and $\lambda=f(u)$, then $\lambda\cdot u^{\otimes m}$ is a best rank-1 approximation of $\mathcal{F}$. Clearly, to solve (5.7), we need to solve two maximization problems:

$$\text{(I)} \quad \max_{x\in\mathbb{S}^{n-1}} f(x), \qquad \text{(II)} \quad \max_{x\in\mathbb{S}^{n-1}} -f(x), \tag{5.8}$$

where $\mathbb{S}^{n-1}:=\{x\in\mathbb{R}^n:\|x\|=1\}$ is the $n-1$ dimensional unit sphere. Suppose $u^+,u^-$ are global maximizers of (I), (II) in (5.8) respectively. By Theorem 5.1.1, if $|f(u^+)|\geq|f(u^-)|$, $f(u^+)\cdot(u^+)^{\otimes m}$ is the best rank-1 approximation; otherwise, $f(u^-)\cdot(u^-)^{\otimes m}$ is the best.

For an introduction to symmetric tensors, we refer to Comon, Golub, Lim and Mourrain [58]. Finding best rank-1 approximations for symmetric tensors is also NP-hard when $m\geq 3$. There exist methods for computing rank-1 approximations for symmetric tensors. When HOPM is directly applied, it is often unreliable for attaining a good symmetric rank-1 approximation, as pointed out in [37]. To get good symmetric rank-1 approximations, Kofidis and Regalia [31] proposed a symmetric higher-order power method (SHOPM), Zhang, Ling and Qi [80] proposed a modified power method. These methods can be easily implemented. Like

for nonsymmetric tensors, they often generate a locally optimal rank-1 approximation or even just a stationary point. Even for the lucky cases that a globally optimal rank-1 approximation is found, these methods typically have difficulty to verify its global optimality. The problem (5.7) is related to extreme $Z$-eigenvalues of symmetric tensors. Recently, Hu, Huang and Qi [24] proposed a method for computing extreme $Z$-eigenvalues for symmetric tensors with even orders. It is to solve a sequence of semidefinite relaxations based on sum of squares representations.

## 5.2 Semidefinite Relaxation Algorithms

To find best rank-1 tensor approximations is equivalent to solving some homogeneous polynomial optimization problems with sphere constraints. In this section, we show how to solve them by using semidefinite relaxations based on sum of squares representations, and study their properties.

### 5.2.1 Symmetric Tensors of Even Orders

Let $\mathcal{F} \in \mathbb{S}^m(\mathbb{R}^n)$ with $m = 2d$ even. To get a best rank-1 approximation of $\mathcal{F}$, we need to solve (5.7), i.e., to maximize $|f(x)|$ over the unit sphere. For this purpose, we need to find the maximum and minimum of $f(x)$ over $\mathbb{S}^{n-1}$.

First, we consider the maximization problem:

$$f_{\max} := \max \quad f(x) \quad \text{s.t.} \quad x^T x = 1. \tag{5.1}$$

The form $f$ is in $x := (x_1, \ldots, x_n)$, determined by the tensor $\mathcal{F}$ as

$$f(x) = \sum_{1 \le i_1, \ldots, i_m \le n} \mathcal{F}_{i_1, \ldots, i_m} x_{i_1} \cdots x_{i_m}. \tag{5.2}$$

Let $[x^d]$ be the monomial vector:

$$[x^d] := \begin{bmatrix} x_1^d & x_1^{d-1}x_2 & \cdots & x_1^{d-1}x_n & \cdots & x_n^d \end{bmatrix}^T.$$

Its length is $\binom{n+d-1}{d}$. The outer product $[x^d][x^d]^T$ is a symmetric matrix with entries being monomials of degree $m$. Let $A_\alpha$ be symmetric matrices such that

$$[x^d][x^d]^T = \sum_{\alpha \in \mathbb{N}^n_m} A_\alpha x^\alpha.$$

For $y \in \mathbb{R}^{\mathbb{N}^n_m}$, define the matrix-valued function $M(y)$ as

$$M(y) := \sum_{\alpha \in \mathbb{N}^n_m} A_\alpha y_\alpha.$$

Then $M(y)$ is a linear pencil in $y$ (i.e., $M(y)$ is a linear matrix-valued function in $y$). Let $f_\alpha, g_\alpha$ be the coefficients such that

$$f := \sum_{\alpha \in \mathbb{N}^n_m} f_\alpha x^\alpha, \quad g := (x^T x)^d = \sum_{\alpha \in \mathbb{N}^n_m} g_\alpha x^\alpha.$$

For $y \in \mathbb{R}^{\mathbb{N}^n_m}$, define

$$\langle f, y \rangle := \sum_{\alpha \in \mathbb{N}^n_m} f_\alpha y_\alpha, \quad \langle g, y \rangle := \sum_{\alpha \in \mathbb{N}^n_m} g_\alpha y_\alpha.$$

A semidefinite relaxation of (5.1) is (cf. [50, 55])

$$f^{\rm sdp}_{\max} := \max_{y \in \mathbb{R}^{\mathbb{N}^n_m}} \quad \langle f, y \rangle \quad \text{s.t.} \quad M(y) \succeq 0, \quad \langle g, y \rangle = 1. \tag{5.3}$$

It can be shown that the the dual of the above is

$$\min \quad \gamma \quad \text{s.t.} \quad \gamma g - f \in \Sigma_{n,m}. \tag{5.4}$$

In the above, $\Sigma_{n,m}$ denotes the cone of SOS forms of degree $m$ and in variables $x_1, \ldots, x_n$. Clearly, $f^{\rm sdp}_{\max} \geq f_{\max}$. When $f^{\rm sdp}_{\max} = f_{\max}$, we say that the semidefinite relaxation (5.3) is tight.

The dual (5.4) is a linear optimization problem with SOS type constraints. Recently, Hu, Huang and Qi [24] proposed an SOS relaxation method for computing the largest or smallest Z-eigenvalues for symmetric tensors of even orders. Since not every nonnegative form is SOS, they consider a sequence of nesting SOS relaxations. The problem (5.4) is equivalent to the lowest order relaxation in [24, Section 4]. In practice, the relaxation (5.4) is often tight, and it is frequently used

because of its simplicity. The SOS relaxation (5.4) was also proposed in [50, Section 1] for optimizing forms over unit spheres. Its approximation quality was also analyzed in [50].

The feasible set of (5.3) is compact, because

$$\mathrm{Trace}(M(y)) \leq \langle g, y\rangle = 1.$$

So (5.3) always has a maximizer, say, $y^*$. If $\mathrm{rank}\, M(y^*) = 1$, then (5.3) is a tight relaxation. In such case, there exists $v^+ \in \mathbb{S}^{n-1}$ such that $y^* = [(v^+)^m]$, because of the structure of $M(y)$. Then, it holds that

$$f(v^+) = \langle f, y^*\rangle = f_{\max}^{\mathrm{sdp}} \geq f_{\max}.$$

This implies that $v^+$ is a global maximizer of (5.1). The vector $v^+$ can be chosen numerically as follows. Let $s \in [n]$ be the index such that

$$y^*_{2de_s} = \max_{1\leq i\leq n} y^*_{2de_i}.$$

Then choose $v^+$ as the normalization:

$$\hat{u} = (y^*_{(2d-1)e_s+e_1}, \ldots, y^*_{(2d-1)e_s+e_n}), \quad v^+ = \hat{u}/\|\hat{u}\|. \tag{5.5}$$

If $\mathrm{rank}(M(y^*)) > 1$ but $M(y^*)$ satisfies a further rank condition, then (5.3) is also tight (cf. [55, Section 2.2]). In computations, no matter (5.3) is tight or not, the vector $v^+$ selected as in (5.5) can be used as an approximation for a maximizer of (5.1).

Second, we consider the minimization problem:

$$f_{\min} := \min \quad f(x) \quad \text{s.t.} \quad x^T x = 1. \tag{5.6}$$

Similarly, a semidefinite relaxation of (5.6) is

$$f_{\min}^{\mathrm{sdp}} := \min_{z\in\mathbb{R}^{\mathbb{N}^n_m}} \quad \langle f, z\rangle \quad \text{s.t.} \quad M(z) \succeq 0, \quad \langle g, z\rangle = 1. \tag{5.7}$$

Its dual optimization problem can be shown to be

$$\max \quad \eta \quad \text{s.t.} \quad f - \eta g \in \Sigma_{n,m}. \tag{5.8}$$

Let $z^*$ be a minimizer of (5.7). Similarly, if $\text{rank}(M(z^*)) = 1$, then (5.7) is a tight relaxation. In such case, a global minimizer $v^-$ can be found as follows: let $t \in [n]$ be the index such that

$$z^*_{2de_t} = \max_{1 \le i \le n} z^*_{2de_i},$$

then choose $v^-$ as the normalization:

$$\tilde{u} = (z^*_{(2d-1)e_t+e_1}, \ldots, z^*_{(2d-1)e_t+e_n}), \quad v^- = \tilde{u}/\|\tilde{u}\|. \tag{5.9}$$

When $\text{rank}\, M(z^*) > 1$, (5.7) might not be a tight relaxation. In computations, the vector $v^-$ can be used as an approximation for a minimizer of (5.6).

Combining the above and inspired by Theorem 5.1.1, we get the following algorithm.

**Algorithm 5.2.1.** (Rank-1 Approximations for Even Symmetric Tensors)
**Input:** *A symmetric tensor $\mathcal{F} \in \mathbb{S}^m(\mathbb{R}^n)$ with an even order $m = 2d$.*
**Output:** *A rank-1 tensor $\lambda \cdot u^{\otimes m}$, with $\lambda \in \mathbb{R}$ and $u \in \mathbb{S}^{n-1}$.*
**Procedure:**

***Step 1*** *Solve the semidefinite relaxation (5.3) and get an optimizer $y^*$.*

***Step 2*** *Choose $v^+$ as in (5.5). If $\text{rank}\, M(y^*) = 1$, let $u^+ = v^+$; otherwise, apply a nonlinear optimization method to get a better solution $u^+$ of (5.1), by using $v^+$ as a starting point. Let $\lambda^+ = f(u^+)$.*

***Step 3*** *Solve the semidefinite relaxation (5.7) and get an optimizer $z^*$.*

***Step 4*** *Choose $v^-$ as in (5.9). If $\text{rank}\, M(z^*) = 1$, let $u^- = v^-$; otherwise, apply a nonlinear optimization method to get a better solution $u^-$ of (5.6), by using $v^-$ as a starting point. Let $\lambda^- = f(u^-)$.*

***Step 5*** *If $|\lambda^+| \ge |\lambda^-|$, output $(\lambda, u) := (\lambda^+, u^+)$; otherwise, output $(\lambda, u) := (\lambda^-, u^-)$.*

*Remark:* In Algorithm 5.2.1, if $\text{rank}\, M(y^*) = \text{rank}\, M(z^*) = 1$, then the output $\lambda \cdot u^{\otimes m}$ is a best rank-1 approximation of $\mathcal{F}$. If this rank condition is not satisfied, then

$\lambda \cdot u^{\otimes m}$ might not be best. The approximation qualities of semidefinite relaxations (5.3) and (5.7) are analyzed in [50, Section 2].

When $m = 2$ or $(n, m) = (3, 4)$, the semidefinite relaxations (5.3) and (5.7) are always tight. This is because every bivariate, or ternary quartic, nonnegative form is always SOS (cf. [65]). For other cases of $(n, m)$, this is not necessarily true. However, this does not occur very often in applications. In our numerical experiments, the semidefinite relaxations (5.3) and (5.7) are often tight.

We want to know when the ranks of $M(y^*)$ and $M(z^*)$ are equal to one. Clearly, when they have rank one, the relaxations (5.3) and (5.7) must be tight. Interestingly, the reverse is often true, although it might be false sometimes (for very few cases). This fact was observed in the field of polynomial optimization. However, we are not able to find suitable references for explaining this fact. Here, we give a natural geometric interpretation for this phenomena, for lack of suitable references. Let $\partial\Sigma_{n,m}$ be the boundary of the cone $\Sigma_{n,m}$.

**Theorem 5.2.2.** *Let $f_{\max}, f_{\max}^{\mathrm{sdp}}, f_{\min}, f_{\min}^{\mathrm{sdp}}, y^*, z^*$ be as above.*

*(i) Suppose $f_{\max} = f_{\max}^{\mathrm{sdp}}$. If $f_{\max} \cdot g - f$ is a smooth point of $\partial\Sigma_{n,m}$, then $\mathrm{rank}\, M(y^*) = 1$.*

*(ii) Suppose $f_{\min} = f_{\min}^{\mathrm{sdp}}$. If $f - f_{\min} \cdot g$ is a smooth point of $\partial\Sigma_{n,m}$, then $\mathrm{rank}\, M(z^*) = 1$.*

*Proof.* (i) Let $\mu$ be the uniform probability measure on the unit sphere $\mathbb{S}^{n-1}$, and let $y^\mu := \int [x^m] d\mu \in \mathbb{R}^{\mathbb{N}^n_m}$. Then, we can show that $M(y^\mu) \succ 0$ and $\langle g, y^\mu \rangle = 1$. This shows that $y^\mu$ is an interior point of (5.3). So, the strong duality holds, that is, the optimal values of (5.3) and (5.4) are equal, and (5.4) achieves the optimal value $f_{\max}^{\mathrm{sdp}}$. The form $\sigma := f_{\max} \cdot g - f$ belongs to $\Sigma_{n,m}$, because $f_{\max} = f_{\max}^{\mathrm{sdp}}$. Let $u$ be a maximizer of $f$ on $\mathbb{S}^{n-1}$. Then, $\sigma(u) = 0$. So, $\sigma$ lies on the boundary $\partial\Sigma_{n,m}$. Let $\hat{y} = [u^m]$. Denote by $\mathbb{R}[x]_m$ the space of all forms in $x$ and of degree $m$. Then

$$\mathcal{H}_{\hat{y}} := \{p \in \mathbb{R}[x]_m : \langle p, \hat{y} \rangle = 0\}$$

is a supporting hyperplane of $\Sigma_{n,m}$ through $\sigma$, because

$$\langle p, \hat{y} \rangle = p(u) \geq 0, \quad \forall\, p \in \Sigma_{n,m}$$

and $\langle \sigma, \hat{y} \rangle = \sigma(u) = 0$. Because $M(y^*) \succeq 0$ and $\langle p, y^* \rangle \geq 0$ for all $p \in \Sigma_{n,m}$, the hyperplane

$$\mathcal{H}_{y^*} := \{p \in \mathbb{R}[x]_m : \langle p, y^* \rangle = 0\}$$

also supports $\Sigma_{n,m}$ through $\sigma$. Since $\sigma$ is a smooth point of $\partial\Sigma_{n,m}$, there is a unique supporting hyperplane $\mathcal{H}$ through $\sigma$. So, $y^* = \hat{y} = [u^m]$. This implies that $M(y^*) = M([u^m]) = [u^d][u^d]^T$ has rank one, where $d = m/2$.

(ii) The proof is the same as for (i). □

By Theorem 5.2.2, when semidefinite relaxations (5.3) and (5.7) are tight, we often have $\operatorname{rank} M(y^*) = \operatorname{rank} M(z^*) = 1$. This fact was observed in our numerical experiments. The reason is that the set of nonsmooth points of the boundary $\partial\Sigma_{n,m}$ has a strictly smaller dimension than $\partial\Sigma_{n,m}$.

### 5.2.2 Symmetric Tensors of Odd Orders

Let $\mathcal{F} \in \mathbb{S}^m(\mathbb{R}^n)$ with $m = 2d-1$ odd. To get a best rank-1 approximation of $\mathcal{F}$, we need to solve the optimization problem (5.7). Since the form $f$, defined as in (5.2), has the odd degree $m$, (5.7) is equivalent to (5.1). We can not directly apply a semidefinite relaxation to solve (5.1). For this purpose, we use a trick that is introduced in [50, Section 4.2].

Let $f_{\max}, f_{\min}$ be as in (5.1), (5.6) respectively. Since $f$ is an odd form,

$$f_{\max} = -f_{\min} \geq 0.$$

Let $x_{n+1}$ be a new variable, in addition to $x = (x_1, \ldots, x_n)$. Let

$$\tilde{x} := (x_1, \ldots, x_n, x_{n+1}), \quad \tilde{f}(\tilde{x}) := f(x)x_{n+1}.$$

Then $\tilde{f}(\tilde{x})$ is a form of even degree $2d$. Consider the optimization problem:

$$\tilde{f}_{\max} := \max_{\tilde{x} \in \mathbb{R}^{n+1}} \quad \tilde{f}(\tilde{x}) \quad \text{s.t.} \quad \|\tilde{x}\| = 1. \tag{5.10}$$

As shown in [50, Section 4.2], it holds that

$$f_{\max} = \sqrt{2d-1}\big(1 - \frac{1}{2d}\big)^{-d} \tilde{f}_{\max}.$$

Since $\tilde{f}$ is an even form, the semidefinite relaxation for (5.10) is

$$\tilde{f}_{\max}^{\text{sdp}} := \max_{y \in \mathbb{N}_{2d}^{n+1}} \quad \langle \tilde{f}, y \rangle \quad \text{s.t.} \quad M(y) \succeq 0, \quad \langle g, y \rangle = 1. \tag{5.11}$$

The vector $y$ is indexed by $(n+1)$-dimensional integer vectors.

Let $y^*$ be a maximizer of (5.11), which always exists because the feasible set of (5.11) is compact. If $\operatorname{rank} M(y^*) = 1$, then (5.11) is a tight relaxation, and a global maximizer $v^+$ of (5.10) can be chosen as in (5.5). (Note that the $n$ in (5.5) should be replaced by $n+1$.) Write $v^+$ as

$$v^+ = (\hat{v}, \hat{t}), \quad \|\hat{v}\|^2 + \hat{t}^2 = 1.$$

If $\hat{v} = 0$ or $\hat{t} = 0$, then $\tilde{f}_{\max} = f_{\max} = 0$ and the zero tensor is the best rank-1 approximation of $\mathcal{F}$. So, we consider the general case $0 < |\hat{t}| < 1$. Note that $\operatorname{sign}(\hat{t}) \cdot \hat{v}$ is a global maximizer of $f$ on the sphere $\|\hat{v}\|_2^2 = 1 - \hat{t}^2$. Let

$$\hat{u} = \operatorname{sign}(\hat{t}) \cdot \hat{v} / \sqrt{1 - \hat{t}^2}. \tag{5.12}$$

Then $\hat{u}$ is a global maximizer of $f$ on $\mathbb{S}^{n-1}$. When $\operatorname{rank} M(y^*) > 1$, the above $\hat{u}$ might not be a global maximizer, but it can be used as an approximation for a maximizer.

Combining the above, we get the following algorithm.

**Algorithm 5.2.3.** (Rank-1 Approximations for Odd Symmetric Tensors)
**Input:** *A symmetric tensor $\mathcal{F} \in \mathbb{S}^m(\mathbb{R}^n)$ with an odd order $m = 2d-1$.*
**Output:** *A rank-1 symmetric tensor $\lambda \cdot u^{\otimes m}$ with $\lambda \in \mathbb{R}$ and $u \in \mathbb{S}^{n-1}$.*
**Procedure:**

***Step 1*** *Solve the semidefinite relaxation (5.11) and get an optimizer $y^*$.*

***Step 2*** *Choose $v^+$ as in (5.5) and $\hat{u}$ as in (5.12). (The $n$ in (5.5) should be replaced by $n+1$.)*

***Step 3*** *If $\operatorname{rank} M(y^*) = 1$, let $u = \hat{u}$; otherwise, apply a nonlinear optimization method to get a better solution $u$ of (5.1), by using $\hat{u}$ as a starting point. Let $\lambda = f(u)$. Output $(\lambda, u)$.*

*Remark:* In Algorithm 5.2.3, if $\operatorname{rank} M(y^*) = 1$, then the output $\lambda \cdot u^{\otimes m}$ is a best rank-1 approximation of the tensor $\mathcal{F}$. If $\operatorname{rank} M(y^*) > 1$, then $\lambda \cdot u^{\otimes m}$ is not necessarily the best. The approximation quality of the semidefinite relaxation (5.11) is analyzed in [50, Section 4]. In our numerical experiments, we often have $\operatorname{rank} M(y^*) = 1$. A similar version of Theorem 5.2.2 is true for Algorithm 5.2.3.

### 5.2.3 Nonsymmetric Tensors

Let $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$ be a nonsymmetric tensor of order $m$. Let $F(x^1, \ldots, x^m)$ be the multi-linear form defined as in (5.4). To get a best rank-1 approximation of $\mathcal{F}$ is equivalent to solving the multilinear optimization problem (5.5). Here we show how to solve it by using semidefinite relaxations.

Without loss of generality, assume $n_m = \max_j n_j$. Since $F(x^1, \ldots, x^m)$ is linear in $x^m$, we can write it as

$$F(x^1, \ldots, x^m) = \sum_{j=1}^{n_m} (x^m)_j F_j(x^1, \ldots, x^{m-1}),$$

where each $F_j(x^1, \ldots, x^{m-1})$ is also a multi-linear form. Let $m' = m - 1$, and

$$F^{sq} := \sum_{j=1}^{n_m} F_j(x^1, \ldots, x^{m'})^2.$$

By the Cauchy-Schwartz inequality, it holds that

$$|F(x^1, \ldots, x^m)| \le F^{sq}(x^1, \ldots, x^{m'})^{1/2} \|x^m\|.$$

The equality occurs in the above if and only if $x^m$ is proportional to

$$\big(F_1(x^1, \ldots, x^{m'}), \ldots, F_{n_m}(x^1, \ldots, x^{m'})\big).$$

Therefore, (5.5) is equivalent to

$$\begin{cases} F_{\max} : \; = \max\limits_{x^1, \ldots, x^{m'}} & F^{sq}(x^1, \ldots, x^{m'}) \\ \text{s.t.} & \|x^1\| = \cdots = \|x^{m'}\| = 1. \end{cases} \tag{5.13}$$

Now we present the semidefinite relaxations for solving (5.13). The outer product

$$\mathcal{K}(x) := x^1 \otimes \cdots \otimes x^{m'}$$

is a vector of length $n_1 \cdots n_{m'}$. Denote

$$\Omega := \{(\imath, \jmath) : \imath, \jmath \in [n_1] \times \cdots \times [n_{m'}]\}.$$

Expand the outer product of $\mathcal{K}(x)$ as

$$\mathcal{K}(x)\mathcal{K}(x)^T = \sum_{(\imath,\jmath)\in\Omega} B_{\imath,\jmath} (x^1)_{\imath_1}(x^1)_{\jmath_1} \cdots (x^{m'})_{\imath_{m'}}(x^{m'})_{\jmath_{m'}},$$

where each $B_{\imath,\jmath}$ is a constant symmetric matrix. For $w \in \mathbb{R}^\Omega$, define

$$K(w) := \sum_{(\imath,\jmath)\in\Omega} B_{\imath,\jmath} w_{\imath,\jmath}.$$

Clearly, $K(w)$ is a linear pencil in $w \in \mathbb{R}^\Omega$. Write

$$F^{sq} = \sum_{(\imath,\jmath)\in\Omega} G_{\imath,\jmath} (x^1)_{\imath_1}(x^1)_{\jmath_1} \cdots (x^{m'})_{\imath_{m'}}(x^{m'})_{\jmath_{m'}},$$

$$h := \|x^1\|^2 \cdots \|x^{m'}\|^2 = \sum_{(\imath,\jmath)\in\Omega} h_{\imath,\jmath} (x^1)_{\imath_1}(x^1)_{\jmath_1} \cdots (x^{m'})_{\imath_{m'}}(x^{m'})_{\jmath_{m'}}.$$

For $w \in \mathbb{R}^\Omega$, we denote

$$\langle F^{sq}, w\rangle := \sum_{(\imath,\jmath)\in\Omega} G_{\imath,\jmath}\, w_{\imath,\jmath}, \qquad \langle h, w\rangle := \sum_{(\imath,\jmath)\in\Omega} h_{\imath,\jmath}\, w_{\imath,\jmath}.$$

A semidefinite relaxation of (5.13) is

$$F^{\text{sdp}}_{\max} := \max \quad \langle F^{sq}, w\rangle \quad \text{s.t.} \quad K(w) \succeq 0, \quad \langle h, w\rangle = 1. \tag{5.14}$$

Define $\Sigma_{n_1,\ldots,n_{m'}}$ to be the cone as

$$\Sigma_{n_1,\ldots,n_{m'}} = \left\{ L \;\middle|\; \begin{array}{c} L = L_1^2 + \cdots + L_k^2 \text{ where each } L_i \\ \text{is a multilinear form in } (x^1, \ldots, x^{m'}) \end{array} \right\}.$$

It can be shown that the dual problem of (5.14) is

$$\min \quad \gamma \quad \text{s.t.} \quad \gamma h - F^{sq} \in \Sigma_{n_1,\ldots,n_{m'}}. \tag{5.15}$$

Clearly, we always have $F^{\text{sdp}}_{\max} \geq F_{\max}$. When the equality occurs, we say that (5.14) is a tight relaxation.

The feasible set of (5.14) is compact, because

$$\mathrm{Trace}(K(w)) = \langle h, w\rangle = 1.$$

Let $w^*$ be a maximizer of (5.14). Like for the case of symmetric tensors, if $\mathrm{rank}\, K(w^*) = 1$, then (5.14) is tight, and there exist vectors $v^1, \ldots, v^{m'}$ of unit length such that $w^* = (v^1(v^1)^T) \otimes \cdots \otimes (v^{m'}(v^{m'})^T)$ and $(v^1, \ldots, v^{m'})$ is a maximizer of (5.13). They can be constructed as follows. Let $\ell \in [n_1] \times \cdots \times [n_{m'}]$ be the index such that

$$w^*_{\ell,\ell} = \max_{(\imath,\imath)\in\Omega} w^*_{\imath,\imath}.$$

Then choose $v^j$ $(j = 1, \ldots, m')$ as:

$$\hat{v}^j = \Big(w^*_{\hat{\ell}_1,\ell}, w^*_{\hat{\ell}_2,\ell}, \ldots, w^*_{\hat{\ell}_{n_j},\ell}\Big), \qquad v^j = \hat{v}^j/\|\hat{v}^j\|, \tag{5.16}$$

where $\hat{\ell}_k = \ell + (k - \ell_j)\cdot e_j$ for each $k \in [n_j]$. When $\mathrm{rank}\, K(w^*) > 1$, the tuple $(v^1, \ldots, v^{m'})$ as in (5.16) might not be a global maximizer of (5.13). But it can be used as an approximation for a maximizer of (5.13).

Combining the above, we get the following algorithm.

**Algorithm 5.2.4.** (Rank-1 Approximations for Nonsymmetric Tensors)
**Input:** *A nonsymmetric tensor* $\mathcal{F} \in \mathbb{R}^{n_1\times\ldots\times n_m}$.
**Output:** *A rank-1 tensor* $\lambda\cdot(u^1 \otimes \cdots \otimes u^m)$ *with* $\lambda \in \mathbb{R}$ *and each* $u^j \in \mathbb{S}^{n_j-1}$.
**Procedure:**

***Step 1*** *Solve the semidefinite relaxation (5.14) and get a maximizer* $w^*$.

***Step 2*** *Choose* $(v^1, \ldots, v^{m'})$ *as in (5.16). Then let*

$$\hat{v}^m := (F_1(v^1, \ldots, v^{m'}), \ldots, F_{n_m}(v^1, \ldots, v^{m'})),$$

*and* $v^m := \hat{v}^m/\|\hat{v}^m\|$.

***Step 3*** *If* $\mathrm{rank}\, K(w^*) = 1$, *let* $u^i = v^i$ *for* $i = 1, \ldots, m$; *otherwise, apply a nonlinear optimization method to get a better solution* $(u^1, \ldots, u^m)$ *of (5.5), by using* $(v^1, \ldots, v^m)$ *as a starting point.*

***Step 4*** *Let* $\lambda = F(u^1, \ldots, u^m)$, *and output* $(\lambda, u^1, \ldots, u^m)$.

*Remark:* In Algorithm 5.2.4, if $\operatorname{rank} K(w^*) = 1$, then the output $\lambda \cdot u^1 \otimes \cdots \otimes u^m$ is a best rank-1 approximation of $\mathcal{F}$. If $\operatorname{rank} K(w^*) > 1$, then $\lambda \cdot u^1 \otimes \cdots \otimes u^m$ is not necessarily the best. The approximation quality of the semidefinite relaxation (5.14) is analyzed in [50, Section 3].

We want to know when $\operatorname{rank} K(w^*) = 1$. Clearly, for this to be true, the relaxation (5.14) must be tight, i.e., $F_{\max} = F_{\max}^{\text{sdp}}$. Like for the symmetric case, the reverse is also often true, as shown in the following theorem. Let $\partial\Sigma_{n_1,\ldots,n_{m'}}$ be the boundary of the cone $\Sigma_{n_1,\ldots,n_{m'}}$.

**Theorem 5.2.5.** *Let $F_{\max}, F_{\max}^{\text{sdp}}, w^*$ be as above. Suppose $F_{\max} = F_{\max}^{\text{sdp}}$. If $F_{\max} \cdot h - F^{sq}$ is a smooth point of $\partial\Sigma_{n_1,\ldots,n_{m'}}$, then $\operatorname{rank} K(w^*) = 1$.*

*Proof.* This can be proved in the same way as for Theorem 5.2.2. Let $(u^1, \cdots, u^{m'})$ be a global maximizer of (5.13). Let $\hat{w} \in \mathbb{R}^\Omega$ be the vector such that

$$\hat{w}_{\imath,\jmath} = (u^1)_{\imath_1}(u^1)_{\jmath_1} \cdots (u^m)_{\imath_m}(u^m)_{\jmath_m} \quad \forall\, (\imath,\jmath) \in \Omega.$$

The key point is the observation that $\langle p, \hat{w} \rangle = 0$ defines a unique supporting hyperplane of $\Sigma_{n_1,\ldots,n_{m'}}$ through $F_{\max} \cdot h - F^{sq}$, when it is a smooth point of the boundary $\partial\Sigma_{n_1,\ldots,n_{m'}}$. The proof proceeds same as for Theorem 5.2.2. □

## 5.3 Numerical Experiments

In this section, we present numerical experiments of using semidefinite relaxations to find best rank-1 tensor approximations. The computations are implemented in Matlab 7.10 on a Dell Linux Desktop with 8GB memory and Intel(R) CPU 2.8GHz. In applications, the resulting semidefinite programs are often large scale. Interior point methods are not very suitable for solving such big semidefinite programs. We use the software `SDPNAL` [83] by Zhao, Sun and Toh, which is based on the Newton-CG Augmented Lagrangian method [81]. In our computations, the default values of the parameters in `SDPNAL` are used. In Algorithms 5.2.1, 5.2.3 and 5.2.4, if the matrices $M(y^*), M(z^*), K(w^*)$ do not have rank one, we apply the nonlinear program solver `fmincon` in Matlab Optimization Toolbox to improve the rank-1 approximations obtained from semidefinite relaxations. Our numerical

experiments show that these algorithms are often able to get best rank-1 approximations and `SDPNAL` is efficient in solving such large scale semidefinite relaxations.

We report the consumed computer time in the format `hr:mn:sc` with `hr` (resp., `mn, sc`) standing for the consumed hours (resp., minutes, seconds). In our presented tables, `min` (resp., `med, max`) stands for the minimum (resp., median, maximum) of quantities like time, errors.

In our computations, the rank of a matrix $A$ is numerically measured as follows: if the singular values of $A$ are $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_t > 0$, then $\operatorname{rank}(A)$ is set to be the smallest $r$ such that $\sigma_{r+1}/\sigma_r < 10^{-6}$. In the display of our computational results, only four decimal digits are shown.

### 5.3.1 Symmetric Tensor Best Rank-1 Approximation

In this subsection, we report numerical experiments for symmetric tensors. We apply Algorithm 5.2.1 for even symmetric tensors, and apply Algorithm 5.2.3 for odd symmetric tensors. In Algorithm 5.2.1, if

$$\operatorname{rank} M(y^*) = \operatorname{rank} M(z^*) = 1, \tag{5.17}$$

then the output tensor $\lambda \cdot u^{\otimes m}$ is a best rank-1 approximation. If $\operatorname{rank} M(y^*) > 1$ or $\operatorname{rank} M(z^*) > 1$, $\lambda \cdot u^{\otimes m}$ is not guaranteed to be a best rank-1 approximation. However, the quantity

$$f_{\text{ubd}} := \max\{|f^{\text{sdp}}_{\max}|, |f^{\text{sdp}}_{\min}|\}$$

is always an upper bound of $|f(x)|$ on $\mathbb{S}^{n-1}$. No matter whether (5.17) holds or not, the error

$$\texttt{aprxerr} := \big||f(u)| - f_{\text{ubd}}\big| / \max\{1, f_{\text{ubd}}\} \tag{5.18}$$

is a measure of the approximation quality of $\lambda \cdot u^{\otimes m}$. When Algorithm 5.2.3 is applied for odd symmetric tensors, $f_{\text{ubd}}$ and `aprxerr` are defined similarly. As in Qi [64], we define the *best rank-*1 *approximation ratio* of a tensor $\mathcal{F} \in \mathbb{S}^m(\mathbb{R}^n)$ as

$$\rho(\mathcal{F}) := \max_{\mathcal{X} \in \mathbb{S}^m(\mathbb{R}^n), \operatorname{rank} \mathcal{X} = 1} \frac{|\langle \mathcal{F}, \mathcal{X} \rangle|}{\|\mathcal{F}\| \|\mathcal{X}\|}. \tag{5.19}$$

If $\lambda \cdot u^{\otimes m}$, with $\|u\| = 1$ and $\lambda = f(u)$, is a best rank-1 approximation of $\mathcal{F}$, then $\rho(\mathcal{F}) = |\lambda|/\|\mathcal{F}\|$. Estimates for $\rho(\mathcal{F})$ are given in Qi [64].

**Example 5.3.1.** ([37, Example 2]) Consider the tensor $\mathcal{F} \in \mathtt{S}^3(\mathbb{R}^2)$ with entries:

$$\mathcal{F}_{111} = 1.5578, \quad \mathcal{F}_{222} = 1.1226, \quad \mathcal{F}_{112} = -2.4443, \quad \mathcal{F}_{221} = -1.0982.$$

When Algorithm 5.2.3 is applied, we get the rank-1 tensor $\lambda \cdot u^{\otimes 3}$ with

$$\lambda = 3.1155, \quad u = (0.9264, -0.3764).$$

It takes about 0.2 second. The computed matrix $M(y^*)$ has rank one. So, we know $\lambda \cdot u^{\otimes 3}$ is a best rank-1 approximation. The error `aprxerr` = 7.3e-9, the ratio $\rho(\mathcal{F}) = 0.6203$, the residual $\|\mathcal{F} - \lambda \cdot u^{\otimes 3}\| = 3.9399$, and $\|\mathcal{F}\| = 5.0228$.

**Example 5.3.2.** ([34, Example 3.6], [80, Example 4.2]) Consider the tensor $\mathcal{F} \in \mathtt{S}^3(\mathbb{R}^3)$ with entries:

$$\mathcal{F}_{111} = -0.1281, \mathcal{F}_{112} = 0.0516, \mathcal{F}_{113} = -0.0954, \mathcal{F}_{122} = -0.1958, \mathcal{F}_{123} = -0.1790,$$
$$\mathcal{F}_{133} = -0.2676, \mathcal{F}_{222} = 0.3251, \mathcal{F}_{223} = \ \ 0.2513, \mathcal{F}_{233} = \ \ 0.1773, \mathcal{F}_{333} = \ \ 0.0338.$$

When Algorithm 5.2.3 is applied, we get the rank-1 tensor $\lambda \cdot u^{\otimes 3}$ with

$$\lambda = 0.8730, \quad u = (-0.3921, 0.7249, 0.5664).$$

It takes about 0.2 second. The computed matrix $M(y^*)$ has rank one, so $\lambda \cdot u^{\otimes 3}$ is a best rank-1 approximation. The error `aprxerr`=1.2e-7, the ratio $\rho(\mathcal{F}) = 0.8890$, the residual $\|\mathcal{F} - \lambda \cdot u^{\otimes 3}\| = 0.4498$ and $\|\mathcal{F}\| = 0.9820$.

**Example 5.3.3.** ([64, Example 2]) Consider the tensor $\mathcal{F} \in \mathtt{S}^3(\mathbb{R}^3)$ with entries:

$$\mathcal{F}_{111} = 0.0517, \mathcal{F}_{112} = 0.3579, \mathcal{F}_{113} = 0.5298, \mathcal{F}_{122} = 0.7544, \mathcal{F}_{123} = 0.2156,$$
$$\mathcal{F}_{133} = 0.3612, \mathcal{F}_{222} = 0.3943, \mathcal{F}_{223} = 0.0146, \mathcal{F}_{233} = 0.6718, \mathcal{F}_{333} = 0.9723.$$

When Algorithm 5.2.3 is applied, we get the rank-1 tensor $\lambda \cdot u^{\otimes 3}$ with

$$\lambda = 2.1110, \quad u = (0.5204, 0.5113, 0.6839).$$

It takes about 0.2 second. Since the computed matrix $M(y^*)$ has rank one, $\lambda \cdot u^{\otimes 3}$ is a best rank-1 approximation. The error `aprxerr`=6.9e-8, the ratio $\rho(\mathcal{F}) = 0.8574$, the residual $\|\mathcal{F} - \lambda \cdot u^{\otimes 3}\| = 1.2672$, and $\|\mathcal{F}\| = 2.4621$.

**Example 5.3.4.** ([34, Example 3.5], [80, Example 4.1]) Consider the tensor $\mathcal{F} \in \mathtt{S}^4(\mathbb{R}^3)$ with entries:

$$\mathcal{F}_{1111} = 0.2883,\ \mathcal{F}_{1112} = -0.0031, \mathcal{F}_{1113} = 0.1973, \mathcal{F}_{1122} = -0.2485, \mathcal{F}_{1123} = -0.2939,$$
$$\mathcal{F}_{1133} = 0.3847,\ \mathcal{F}_{1222} = \ \ 0.2972, \mathcal{F}_{1223} = 0.1862, \mathcal{F}_{1233} = \ \ 0.0919, \mathcal{F}_{1333} = -0.3619,$$
$$\mathcal{F}_{2222} = 0.1241,\ \mathcal{F}_{2223} = -0.3420,\ \mathcal{F}_{2233} = 0.2127,\ \mathcal{F}_{2333} = 0.2727, \mathcal{F}_{3333} = -0.3054.$$

Applying Algorithm 5.2.1, we get $\lambda^+ \cdot (u^+)^{\otimes m}$ and $\lambda^- \cdot (u^-)^{\otimes m}$ with

$$\lambda^+ = 0.8893, \quad u^+ = (-0.6672, -0.2470, 0.7027),$$

$$\lambda^- = -1.0954, \quad u^- = (-0.5915, 0.7467, 0.3043).$$

It takes about 0.3 second. Since $|\lambda^+| < |\lambda^-|$, the output rank-1 tensor is $\lambda \cdot u^{\otimes 4}$ with $\lambda = \lambda^-, u = u^-$. The computed matrices $M(y^*), M(z^*)$ both have rank one, so $\lambda \cdot u^{\otimes 4}$ is a best rank-1 approximation. The error `aprxerr`=2.8e-7, the ratio $\rho(\mathcal{F}) = 0.4863$, the residual $\|\mathcal{F} - \lambda \cdot u^{\otimes 4}\| = 1.9683$, and $\|\mathcal{F}\| = 2.2525$.

**Example 5.3.5.** Consider the tensor $\mathcal{F} \in \mathtt{S}^6(\mathbb{R}^3)$ with

$$\mathcal{F}_{111111} = 2, \mathcal{F}_{111122} = 1/3, \mathcal{F}_{111133} = 2/5, \mathcal{F}_{112222} = 1/3, \mathcal{F}_{112233} = 1/6,$$
$$\mathcal{F}_{113333} = 2/5, \mathcal{F}_{222222} = 2, \mathcal{F}_{222233} = 2/5, \mathcal{F}_{223333} = 2/5, \mathcal{F}_{333333} = 1,$$

and $\mathcal{F}_{i_1,\ldots,i_6} = 0$ if $(i_1, \ldots, i_6)$ is not a permutation of an index in the above. We can verify that

$$f(x) = 2\|x\|^6 - M(x),$$

where $M(x) = x_1^4x_2^2 + x_1^2x_2^4 + x_3^6 - 3x_1^2x_2^2x_3^2$ is the Motzkin polynomial, which is nonnegative everywhere but not SOS (cf. [65]). Since $0 \le M(x) \le \|x\|^6$, we can show that $f_{\max} = 2$, $f_{\min} = 1$. Applying Algorithm 5.2.1, we get

$$f_{\max}^{\text{sdp}} = 2.0046, \quad v^+ = (0, 1, 0), \quad f(v^+) = 2,$$

$$f_{\min}^{\text{sdp}} = 1.0000, \quad v^- = (0, 0, 1), \quad f(v^-) = 1.$$

The matrix $M(z^*)$ has rank one, so $\lambda^- = f(v^-)$ and $u^- = v^-$. The matrix $M(y^*)$ has rank 7, which is bigger than one, so we apply `fmincon` to improve $v^+$ but get

the same point $u^+ = v^+$; let $\lambda^+ = f(u^+)$. Since $|\lambda^-| < |\lambda^+|$, the output rank-1 tensor is $\lambda \cdot u^{\otimes 6}$ with

$$\lambda = 2.0000, \qquad u = (0, 1, 0).$$

Since $f_{\max} = \lambda = f(u)$, we know $\lambda \cdot u^{\otimes 6}$ is a best rank-1 approximation, by Theorem 5.1.1. The best rank-1 approximation ratio $\rho(\mathcal{F}) = 0.4046$.

**Example 5.3.6.** (Random Examples)

We explore the performance of Algorithms 5.2.1 and 5.2.3 on finding best rank-1 approximations for randomly generated symmetric tensors. We generate $\mathcal{F} \in \mathbb{S}^m(\mathbb{R}^n)$ with each entry being a random variable obeying Gaussian distribution (by `randn` in Matlab). For each generated $\mathcal{F}$, the semidefinite relaxations (5.3), (5.7) and (5.11) can be expressed in the standard dual form

$$\begin{cases} \max & b_1\mu_1 + \cdots + b_M\mu_M \\ \text{s.t.} & F_0 - \sum_{i=1}^M \mu_i F_i \succeq 0, \end{cases} \tag{5.20}$$

where $F_i$ are constant symmetric matrices (cf. [76]). In (5.20), let $N$ be the length of matrices $F_i$, and $M$ be the number of variables. For pairs $(n, m)$, if the semidefinite relaxation matrix length $N < 1000$, we test 50 instances of $\mathcal{F}$ randomly; otherwise if $N > 1000$, we test 10 instances of $\mathcal{F}$ randomly. For a range of values of $(n, m)$, the computational results are shown in Table 5.1.

From Table 5.1, we can observe that Algorithms 5.2.1 and 5.2.3 generally produce accurate best rank-1 approximations in a short time. For some very big problems, like 3-tensors of dimension 40, or 4-tensors of dimension 35, we are able to get accurate best rank-1 approximations within a reasonable time. For most instances, we are able to get best rank-1 approximations, because the computed matrices $M(y^*), M(z^*)$ have rank one. For a few instances, their ranks are bigger than one, and the errors `aprxerr` are a bit relatively large, like in the order of $10^{-3}$ or $10^{-2}$. This is probably because the semidefinite relaxations are not very tight.

### 5.3.2 Nonsymmetric Tensor Best Rank-1 Approximation

In this subsection, we present numerical results for nonsymmetric tensors. In Algorithm 5.2.4, if $\operatorname{rank} K(w^*) = 1$, the output $\lambda \cdot u^1 \otimes \cdots \otimes u^m$ is a best rank-1

**Table 5.1**: Computational results in Example 5.3.6.

| $(n,m)$ | (N,M) | time (min,med,max) | | | aprxerr (min,med,max) |
|---|---|---|---|---|---|
| (10,3) | (66,1000) | 0:00:01 | 0:00:01 | 0:00:03 | (7.9e-9, 4.5e-8, 2.9e-6) |
| (20,3) | (231,10625) | 0:00:03 | 0:00:08 | 0:00:13 | (2.4e-9, 3.6e-7, 4.3e-6) |
| (30,3) | (496,46375) | 0:01:14 | 0:01:29 | 0:02:01 | (9.1e-9, 7.4e-7, 1.4e-5) |
| (40,3) | (861,135750) | 0:06:32 | 0:10:04 | 0:13:09 | (1.3e-9, 4.6e-6, 2.3e-3) |
| (50,3) | (1326,316250) | 0:12:39 | 0:13:34 | 0:14:01 | (3.2e-9, 1.3e-6, 2.0e-3) |
| (15,4) | (120,3060) | 0:00:01 | 0:00:03 | 0:00:04 | (4.0e-9, 1.1e-7, 1.3e-6) |
| (20,4) | (210,8854) | 0:00:52 | 0:01:09 | 0:01:25 | (1.2e-8, 1.8e-7, 6.3e-3) |
| (25,4) | (325,20475) | 0:00:30 | 0:00:35 | 0:00:56 | (4.7e-9, 1.3e-7, 1.0e-5) |
| (30,4) | (465,40919) | 0:06:03 | 0:07:36 | 0:09:31 | (1.2e-8, 1.1e-6, 9.6e-4) |
| (35,4) | (630,73815) | 0:02:46 | 0:04:57 | 0:06:54 | (4.1e-8, 1.6e-7, 7.4e-3) |
| (10,5) | (286,8007) | 0:00:08 | 0:00:14 | 0:00:17 | (4.3e-8, 4.1e-7, 4.1e-6) |
| (15,5) | (816,54263) | 0:03:46 | 0:03:58 | 0:07:24 | (4.4e-8, 2.5e-6, 1.1e-3) |
| (20,5) | (1771,230229) | 0:28:14 | 0:30:30 | 0:43:27 | (4.7e-7, 3.7e-6, 5.7e-6) |
| (10,6) | (220,5004) | 0:00:11 | 0:00:14 | 0:00:20 | (1.3e-7, 6.4e-7, 3.5e-2) |
| (15,6) | (680,38759) | 0:03:14 | 0:04:19 | 0:04:53 | (4.8e-8, 2.5e-3, 4.9e-2) |
| (20,6) | (1540,177099) | 0:39:28 | 0:45:39 | 0:54:59 | (2.8e-8, 6.6e-5, 1.0e-2) |

approximation of $\mathcal{F}$. If $\operatorname{rank} K(w^*) > 1$, $\lambda \cdot u^1 \otimes \cdots \otimes u^m$ might not be the best. However, the quantity

$$F_{\text{ubd}} := \sqrt{\left|F_{\max}^{\text{sdp}}\right|}$$

is always an upper bound of $|F(x^1, \ldots, x^m)|$ on $\mathbb{S}^{n_1-1} \times \cdots \times \mathbb{S}^{n_m-1}$. Like in (5.18), we can measure the quality of $\lambda \cdot u^1 \otimes \cdots \otimes u^m$ by the error

$$\texttt{aprxerr} := \left||F(u^1, \cdots, u^m)| - F_{\text{ubd}}\right| / \max\{1, F_{\text{ubd}}\}. \tag{5.21}$$

Like the symmetric case, we define the *best rank*-1 *approximation ratio* of a tensor $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$ as (cf. Qi [64])

$$\rho(\mathcal{F}) := \max_{\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_m}, \operatorname{rank} \mathcal{X} = 1} \frac{|\langle \mathcal{F}, \mathcal{X} \rangle|}{\|\mathcal{F}\| \|\mathcal{X}\|}. \tag{5.22}$$

Clearly, if $\lambda \cdot u^1 \otimes \cdots \otimes u^m$, with each $\|u^i\| = 1$ and $\lambda = F(u^1, \ldots, u^m)$, is a best rank-1 approximation of $\mathcal{F}$, then $\rho(\mathcal{F}) = |\lambda| / \|\mathcal{F}\|$.

**Example 5.3.7.** ([37, Example 3]) Consider the tensor $\mathcal{F} \in \mathbb{R}^{2\times 2\times 2\times 2}$ with

$$\mathcal{F}_{1111} = 25.1, \quad \mathcal{F}_{1212} = 25.6, \quad \mathcal{F}_{2121} = 24.8, \quad \mathcal{F}_{2222} = 23,$$

and the resting entries are zeros. Applying Algorithm 5.2.4, we get the rank-1 tensor $\lambda \cdot u^1 \otimes u^2 \otimes u^3 \otimes u^4$ with

$$\lambda = 25.6000, \quad u^1 = (1,0), \quad u^2 = (0,1), \quad u^3 = (1,0), \quad u^4 = (0,1).$$

It takes about 0.3 second. The matrix $K(w^*)$ has rank one, so $\lambda \cdot u^1 \otimes u^2 \otimes u^3 \otimes u^4$ is a best rank-1 approximation. The error `aprxerr` =8.9e-10, the ratio $\rho(\mathcal{F}) = 0.5194$, the residual $\|\mathcal{F} - \lambda \cdot u^1 \otimes u^2 \otimes u^3 \otimes u^4\| = 42.1195$, and $\|\mathcal{F}\| = 49.2890$.

**Example 5.3.8.** ([64, Example 1]) Consider the tensor $\mathcal{F} \in \mathbb{R}^{3\times3\times3}$ with

$$\begin{aligned}
&\mathcal{F}_{111} = 0.4333, \mathcal{F}_{121} = 0.4278, \mathcal{F}_{131} = 0.4140, \mathcal{F}_{211} = 0.8154, \mathcal{F}_{221} = 0.0199,\\
&\mathcal{F}_{231} = 0.5598, \mathcal{F}_{311} = 0.0643, \mathcal{F}_{321} = 0.3815, \mathcal{F}_{331} = 0.8834, \mathcal{F}_{112} = 0.4866,\\
&\mathcal{F}_{122} = 0.8087, \mathcal{F}_{132} = 0.2073, \mathcal{F}_{212} = 0.7641, \mathcal{F}_{222} = 0.9924, \mathcal{F}_{232} = 0.8752,\\
&\mathcal{F}_{312} = 0.6708, \mathcal{F}_{322} = 0.8296, \mathcal{F}_{332} = 0.1325, \mathcal{F}_{113} = 0.3871, \mathcal{F}_{123} = 0.0769,\\
&\mathcal{F}_{133} = 0.3151, \mathcal{F}_{213} = 0.1355, \mathcal{F}_{223} = 0.7727, \mathcal{F}_{233} = 0.4089, \mathcal{F}_{313} = 0.9715,\\
&\mathcal{F}_{323} = 0.7726, \mathcal{F}_{333} = 0.5526.
\end{aligned}$$

Applying Algorithm 5.2.4, we get the rank-1 tensor $\lambda \cdot u^1 \otimes u^2 \otimes u^3$ with

$$\lambda = 2.8167, \qquad u^1 = (0.4281, 0.6557, 0.6220),$$

$$u^2 = (0.5706, 0.6467, 0.5062), \quad u^3 = (0.4500, 0.7094, 0.5424).$$

It takes less than one second. The matrix $K(w^*)$ has rank one, so $\lambda \cdot u^1 \otimes u^2 \otimes u^3$ is a best rank-1 approximation. The error `aprxerr` = 3.9e-8, the ratio $\rho(\mathcal{F}) = 0.9017$, the residual $\|\mathcal{F} - \lambda \cdot u^1 \otimes u^2 \otimes u^3\| = 1.3510$, and $\|\mathcal{F}\| = 3.1239$.

**Example 5.3.9.** ([46, Section 4.1]) Consider the tensor $\mathcal{F} \in \mathbb{R}^{3\times3\times3}$ with

$$\begin{aligned}
&\mathcal{F}_{111} = \ 0.0072, \mathcal{F}_{121} = -0.4413, \mathcal{F}_{131} = \ 0.1941, \mathcal{F}_{211} = -0.4413, \mathcal{F}_{221} = \ 0.0940,\\
&\mathcal{F}_{231} = \ 0.5901, \mathcal{F}_{311} = \ 0.1941, \mathcal{F}_{321} = -0.4099, \mathcal{F}_{331} = -0.1012, \mathcal{F}_{112} = -0.4413,\\
&\mathcal{F}_{122} = \ 0.0940, \mathcal{F}_{132} = -0.4099, \mathcal{F}_{212} = \ 0.0940, \mathcal{F}_{222} = \ 0.2183, \mathcal{F}_{232} = \ 0.2950,\\
&\mathcal{F}_{312} = \ 0.5901, \mathcal{F}_{322} = \ 0.2950, \mathcal{F}_{332} = \ 0.2229, \mathcal{F}_{113} = \ 0.1941, \mathcal{F}_{123} = \ 0.5901,\\
&\mathcal{F}_{133} = -0.1012, \mathcal{F}_{213} = -0.4099, \mathcal{F}_{223} = \ 0.2950, \mathcal{F}_{233} = \ 0.2229, \mathcal{F}_{313} = -0.1012,\\
&\mathcal{F}_{323} = \ 0.2229, \mathcal{F}_{333} = -0.4891.
\end{aligned}$$

We apply Algorithm 5.2.4, and get an upper bound $F_{\text{ubd}} = 1.0000$. The computed matrix $K(w^*)$ has rank three, so we use `fmincon` to improve the solution and get the rank-1 tensor $\lambda \cdot u^1 \otimes u^2 \otimes u^3$ with

$$\lambda = 1.0000, \qquad u^1 = (0.7955, 0.2491, 0.5524),$$

$$u^2 = (-0.0050, 0.9142, -0.4051), \quad u^3 = (-0.6060, 0.3195, 0.7285).$$

It takes less than one second. Since $\lambda = F(u^1, u^2, u^3) = F_{\text{ubd}}$, we know $\lambda \cdot u^1 \otimes u^2 \otimes u^3$ is a best rank-1 approximation, by Theorem 5.1.1. The error `aprxerr` = 6.0e-9, the ratio $\rho(\mathcal{F}) = 0.5773$, the residual $\|\mathcal{F} - \lambda \cdot u^1 \otimes u^2 \otimes u^3\| = 1.4143$, and $\|\mathcal{F}\| = 1.7321$.

**Example 5.3.10.** Let $B$ be the symmetric matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -1/2 & 0 & 0 & 0 & -1/2 \\ 0 & 2 & 0 & -1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1/2 & 0 & 0 \\ 0 & -1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1/2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1/2 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & -1/2 & 0 \\ 0 & 0 & -1/2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1/2 & 0 & 0 & 0 \\ -1/2 & 0 & 0 & 0 & -1/2 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The eigenvalues of $B$ are

$$\frac{2-\sqrt{5}}{2}, \quad 0, \quad \frac{3}{2}, \quad \frac{2+\sqrt{5}}{2},$$

which are all less than 3. Consider the tensor $\mathcal{F} \in \mathbb{R}^{3\times 3\times 9}$ such that

$$F^{sq}(x^1, x^2) = (x^1 \otimes x^2)^T (3I_9 - B)(x^1 \otimes x^2).$$

The bi-quadratic form $3\|x^1\|_2^2\|x^2\|_2^2 - F^{sq}(x^1, x^2)$ is nonnegative but not SOS (cf. [7]). The minimum of $(x^1 \otimes x^2)^T B (x^1 \otimes x^2)$ over $\mathbb{S}^2 \times \mathbb{S}^2$ is zero ([43, Example 5.1]), so $F_{\max} = 3$. We apply Algorithm 5.2.4. The computed matrix $K(w^*)$ has rank 4, which is bigger than one. The upper bound $F_{\max}^{\text{sdp}} = 3.0972$. Applying `fmincon`, we get the improved tuple $(u^1, u^2, u^3)$ and $\lambda$ as

$$u^1 = (0, 1, 0), \ u^2 = (1, 0, 0), \ u^3 = (0, 0.1246, 0, 0, 0, 0, -0.9922, 0, 0),$$

**Table 5.2**: Computational results for Example 5.3.11 with $m = 3$.

| $(n_1, n_2, n_3)$ | time (min,med,max) | | | aprxerr (min,med,max) |
|---|---|---|---|---|
| $(10 \times 10 \times 10)$ | 0:00:02 | 0:00:02 | 0:00:03 | (1.6e-9,2.2e-8,2.7e-7) |
| $(15 \times 15 \times 15)$ | 0:00:10 | 0:00:12 | 0:00:18 | (5.9e-9,5.8e-7,2.1e-3) |
| $(20 \times 20 \times 20)$ | 0:00:05 | 0:00:48 | 0:01:24 | (1.9e-9,5.7e-7,5.2e-3) |
| $(25 \times 25 \times 25)$ | 0:00:40 | 0:02:26 | 0:04:57 | (3.2e-9,4.6e-7,5.1e-2) |
| $(30 \times 30 \times 30)$ | 0:05:48 | 0:07:57 | 0:11:31 | (3.6e-8,1.6e-3,3.5e-2) |
| $(35 \times 35 \times 35)$ | 0:21:43 | 0:27:04 | 1:00:05 | (1.1e-5,7.7e-3,5.7e-2) |
| $(40 \times 40 \times 40)$ | 1:10:05 | 1:30:24 | 1:36:24 | (4.8e-4,9.4e-4,1.4e-2) |

$$\lambda = F(u^1, u^2, u^3) = 1.7321 = \sqrt{F_{\max}}.$$

So, $\lambda \cdot u^1 \otimes u^2 \otimes u^3$ is a best rank-1 approximation, by Theorem 5.1.1. The ratio $\rho(\mathcal{F}) = 0.4083$, the residual $\|\mathcal{F} - \lambda \cdot u^1 \otimes u^2 \otimes u^3\| = 3.8730$ and $\|\mathcal{F}\| = 4.2426$.

**Example 5.3.11.** (Random Examples)

We explore the performance of Algorithm 5.2.4 on randomly generated nonsymmetric tensors $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_m}$. The entries of $\mathcal{F}$ are generated obeying Gaussian distributions (by `randn` in Matlab). As in (5.20), let $N$ be the length of matrices and $M$ be the number of variables in the semidefinite relaxations. If $N < 1000$, we generate 50 instances of $\mathcal{F}$ randomly; if $N > 1000$, we generate 10 instances of $\mathcal{F}$ randomly. We apply Algorithm 5.2.4 to get rank-1 approximations. The computational results for orders $m = 3, 4, 5$ are shown in Tables 5.2, 5.3, and 5.4 respectively. When $n_1 = \cdots = n_m$, the sizes of the semidefinite relaxations are typically much larger than the sizes for symmetric tensors. For instance, for $(n, m) = (40, 3)$, $(N, M) = (861, 135750)$ for the symmetric case, while $(N, M) = (1600, 672399)$ for the nonsymmetric case. Typically, Algorithm 5.2.4 takes more time than Algorithms 5.2.1 and 5.2.3, when the input tensors have same dimensions and orders.

We can observe from Tables 5.2, 5.3, and 5.4 that for most instances, Algorithm 5.2.4 is able to get best rank-1 approximations very accurately, within a reasonable short time. For a few cases, the errors are a bit relatively large, around $10^{-2}$, which is probably because the semidefinite relaxations are not very tight.

□

**Table 5.3**: Computational results for Example 5.3.11 with $m = 4$.

| $(n_1, n_2, n_3, n_4)$ | time (min,med,max) | | | aprxerr(min,med,max) |
|---|---|---|---|---|
| $(5 \times 5 \times 5 \times 5)$ | 0:00:02 | 0:00:03 | 0:00:05 | (1.0e-10,1.7e-8,3.1e-7) |
| $(8 \times 8 \times 8 \times 8)$ | 0:00:09 | 0:00:17 | 0:00:28 | (2.3e-7,1.5e-6,1.1e-5) |
| $(10 \times 10 \times 10 \times 10)$ | 0:00:57 | 0:01:52 | 0:10:24 | (9.4e-8,2.0e-6,6.6e-3) |
| $(15 \times 15 \times 5 \times 15)$ | 0:02:18 | 0:07:53 | 0:13:54 | (1.8e-7,3.7e-7,3.7e-3) |
| $(12 \times 12 \times 12 \times 12)$ | 0:07:24 | 0:10:47 | 0:39:07 | (1.7e-7,3.3e-6,2.7e-2) |
| $(20 \times 20 \times 5 \times 20)$ | 1:13:46 | 1:25:02 | 1:39:53 | (2.4e-2,3.4e-2,4.9e-2) |

**Table 5.4**: Computational results for Example 5.3.11 with $m = 5$.

| $(n_1, n_2, n_3, n_4, n_5)$ | time (min,med,max) | | | aprxerr (min,med,max) |
|---|---|---|---|---|
| $(5 \times 5 \times 5 \times 5 \times 5)$ | 0:00:14 | 0:00:24 | 0:00:35 | (7.2e-8,3.7e-7,3.6e-6) |
| $(10 \times 5 \times 5 \times 4 \times 10)$ | 0:00:57 | 0:01:20 | 0:03:27 | (9.7e-8,4.7e-7,1.5e-5) |
| $(10 \times 5 \times 8 \times 5 \times 10)$ | 0:31:46 | 0:50:01 | 1:12:14 | (2.1e-8,2.0e-6,5.3e-2) |
| $(8 \times 8 \times 8 \times 4 \times 10)$ | 0:18:30 | 0:51:45 | 0:53:50 | (2.0e-7,6.5e-7,1.8e-2) |

Chapter 5, in full, is a reprint of the material that has been submitted for publication as it may appear in the article "Semidefinite Relaxations on Tensor Best Rank-1 Approximation" by Jiawang Nie and Li Wang, in SIAM Journal on Matrix Analysis and Applications, 2014. The dissertation author was one of the authors of this paper.

# Chapter 6

# Conclusions

In this chapter, we first give a discussion of the numerical issues related to polynomial optimization.

In Section 2.3, we present extensive numerical examples for testing the efficiency of regularization method for large scale polynomial optimization. In Chapter 5, we provide one main application of large scale polynomial optimization, i.e., finding the best rank-1 approximation of tensors. Numerical results show that regularization method performs well on solving large scale polynomial optimization problems. However, SDP relaxations arising from polynomial optimization are harder to solve than general SDP problems. A reason for this is that the polynomials are not scaled very well sometimes. For instance, if the optimal $Z^*$ has rank one, then $Z^* = [x^*]_d[x^*]_d^T$ ($x^*$ is a minimizer) has entries of the form

$$1, x_1^*, \ldots, (x_1^*)^2, \ldots, (x_1^*)^{2d}, \ldots, (x_n^*)^{2d}.$$

Clearly, if some coordinate $x_i^*$ is either small or big, then $Z^*$ is badly scaled and its entries $Z_{ij}^*$ easily suffer from underflow/overflow during the computation. This might cause severe ill-conditioning in computations and make the computed solutions less accurate. Scaling is a useful approach to overcome this issue. In [20, 62], it was also pointed out that scaling is important in solving polynomial optimization efficiently. Generally, there is no simple rule to select the best scaling factor. In [55, Section 5.1], we propose a practical scaling procedure, for cleanness, we will not repeat it here.

Another main issue that may cause bad performance of regularization method is the degeneracy. In [81], it was shown that if the SDP problem is nondegenerate, then regularization method has good convergence; otherwise, it might converge very slowly or even does not converge. Generally, it is difficult to check in advance whether an SDP relaxation is degenerate or not. For SDP relaxation (2.2)-(2.3), a typical case for it to be degenerate is that the polynomial optimization has several distinct global minimizers. To see this for the unconstrained polynomial optimization, suppose it has two distinct global minimizers $u^*, v^*$ and the SDP relaxation (2.2) is exact. Then, the optimal values of (2.2) and (2.3) are equal, and (2.3) has two distinct optimal $Z^*$ (being $[u^*]_d[u^*]_d^T$ and $[v^*]_d[v^*]_d^T$). This implies the primal SDP relaxation (2.2) is degenerate. The situation is similar for constrained polynomial optimization. From this observation, regularization methods might not be very efficient if the SDP relaxation is exact and there are more than one distinct optimizers. It is an interesting topic for future study to solve degenerate large scale semidefinite programming problems efficiently.

Throughout this thesis, we mainly use software `SDPNAL` [83] to solve large scale SDPs, i.e., regularization method to solve large scale polynomial optimization problems. Another method that might be useful in applications is the low rank SDP approach by Burer and Monteiro [6] (implemented in software `SDPLR` [5]). In some cases, the dual optimal $Z^*$ of SDP relaxations might have low rank. Thus, in such situations, `SDPLR` would be applied to solve the dual SDP relaxation like (2.3) or (2.12) (not the primal SDP relaxation (2.2) or (2.11), since $X^*$ typically has high rank). Even if the performance of `SDPLR` is similar to `SDPNAL`, `SDPLR` is less attractive theoretically and suitable only when $Z^*$ has low rank. This is because the basic idea of `SDPLR` is to change SDP into a nonlinear programming problem via matrix factorization, and typically one would only get a local optimal solution. But `SDPLR` can not guarantee that the computed solution is a minimizer of the original SDP relaxation. Even if an optimal solution of SDP relaxations is found, its optimality can not be verified. A reason for this is that `SDPLR` is not a primal-dual type method, and typically a primal-dual pair is required to check optimality. On the other hand, the computational performance of `SDPLR` is promising. It is an

interesting future work to investigate properties of the low rank method in solving polynomial optimization.

In Chapter 5, to find best rank-1 approximations, we only present the lowest order semidefinite relaxations, and we use local method to improve the solution if the semidefinite relaxation is not tight. Actually, higher order semidefinite relaxation can be applied, as Lasserre's SDP relaxation presented in Section 2.1, and we can get a convergent hierarchy of semidefinite relaxations, as proved by Lasserre [39]. Indeed, this hierarchy almost always converges within finitely many steps, as shown in [52]. The size of semidefinite problems increases quickly with relaxation order increasing. And semidefinite relaxations in rank-1 tensor approximations are often large scale. Theoretically, we can increase the relaxation order, however, computationally, it might generate an SDP problem that is really large scale and it can not be solved by current software `SDPNAL`.

In subsection 2.2.2, we prove that the exactness of Jacobian SDP relaxation method [51] can be weakened as having finite singularities. We guess that the exactness of Jacobian SDP relaxation can be further weakened as having finite real singularities, and we are also interested in finding an example that illustrate this fact. The advantage of Jacobian SDP relaxation is that it has finite convergence under some generic conditions. However, in subsection 2.2.3, it is easy to see that defining the redundant polynomials for Jacobian SDP relaxations is complicated, so Jacobian SDP relaxation method is more interesting theoretically. In [51, Section 4], Nie proposed some variations of Jacobian SDP relaxation, which greatly simplified the procedure for defining the redundant polynomials.

# Bibliography


[1] P. Comon A. Bernardi, J. Brachat and B. Mourrain. General tensor decomposition, moment matrices and applications. *J. Symbolic Comput.*, 52:51–71, 2013.

[2] Farid Alizadeh, Jean Pierre A. Haeberly, and Michael L. Overton. Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results. *SIAM Journal On Optimization*, 8(3):746–768, 1998.

[3] Binita Bhattacharjee, William H. Green, and Paul I. Barton. Interval methods for semi-infinite programs. *Computational Optimization and Applications*, 30(1):63–93, 2005.

[4] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *Proc. 3rd IPSN*, pages 46–54, 2004.

[5] S. Burer. SDPLR: a C package for solving large-scale semidefinite programming problems. http://dollar.biz.uiowa.edu/ sburer.

[6] S. Burer and R. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming, Ser. B*, 95:329–357, 2003.

[7] M. D. Choi. Positive semidefinite biquadratic forms. *Linear Algebra and Applications*, 12:95–100, 1975.

[8] P. Comon. Tensor decompositions - state of the art and applications. In *IMA Conf. in signal processing, Warwick, UK*, 2000.

[9] Raúl E. Curto and Lawrence A. Fialkow. Truncated $K$-moment problems in several variables. *Journal of Operator Theory*, 54(1):189–226, 2005.

[10] D.A.Cox, J.B. Little, and D.O'Shea. *Ideals, Varieties, and Algorithms: an Introduction to Computational Algebraic Geometry and Commutative Algebra, Second Edition. Undergraduate Texts in Mathematics.* Springer-Verlag, New York, 1997.

[11] D.Hilbert. *über die Darstellung definiter Formen als Summe von Formenquadraten*, volume 32. Mathematische Annalen, 1888.

[12] Jack Elzinga and Thomas G. Moore. A central cutting plane algorithm for the convex programming problem. *Mathematical Programming*, 8:134–145, 1975.

[13] Israel M. Gelfand, Mikhail Kapranov, and Andrei Zelevinsky. *Discriminants, Resultants, and Multidimensional Determinants*. Mathematics: Theory & Applications. Birkhäuser, 1994.

[14] Keat-Choon Goh, Michael G. Safonov, and George P. Papavassilopoulos. Global optimization for the biaffine matrix inequality problem. *Journal of Global Optimization*, 7(4):365–380, 1995.

[15] Aurelien Greuet, Feng Guo, Mohab Safey El Din, and Lihong Zhi. Global optimization of polynomials restricted to a smooth variety using sums of squares. *Journal of Symbolic Computation*, 47(5):503–518, 2012.

[16] Feng Guo, Li Wang, and Guangming Zhou. Minimizing rational polynomial by exact Jacobian SDP relaxation applicable to finite singularities. *Journal of Global Optimization*, 58(2):261–284, 2014.

[17] R. Saigal H. Wolkowicz and L. Vandenberghe. *Handbook of semidefinite programming*. Kluwer, 2000.

[18] Didier Henrion and Jean B. Lasserre. *Detecting global optimality and extracting solutions in GloptiPoly*, volume 312. Springer, Berlin, 2005.

[19] Didier Henrion and Jean B. Lasserre. Convergent relaxations of polynomial matrix inequalities and static output feedback. *IEEE Transactions on Automatic Control*, 51(2):192–202, 2006.

[20] Didier Henrion, Jean B. Lasserre, and Johan Löfberg. GloptiPoly 3: moments, optimization and semidefinite programming. *Optimization Methods and Software*, 24(4-5):761–779, 2009.

[21] R. Hettich and K. O. Kortanek. Semi-infinite programming: theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[22] C. Hillar and L.-H. Lim. Most tensor problems are NP-hard. *Journal of the ACM*, To appear, 2013.

[23] Camile W. J. Hol and Carsten W. Scherer. Sum of squares relaxations for polynomial semi-definite programming. In *International Symposium on Mathematical Theory of Networks and Systems*, Leuven, Belgium, July 2004.

[24] S. Hu, Z. H. Huang, and L. Qi. Finding the extreme Z-eigenvalues of tensors via a sequential semidefinite programming method. *Numerical Linear Algebra with Applications*, 20:972–984, 2013.

[25] B. Mourrain J. Brachat, P. Comon and E. Tsigaridas. Symmetric tensor decomposition. *Linear Algebra Appl.*, 433:1851–1872, 2010.

[26] F. Rendl J. Malick, J. Povh and A. Wiegele. Regularization methods for semidefinite programming. *SIAM Journal on Optimization*, 20(1):336–356, 2009.

[27] F. Rendl J. Povh and A. Wiegele. A boundary point method to solve semidefinite programs. *Computing*, 78:277–286, 2006.

[28] Michel Coste Jacek Bochnak and Marie-Francoise Roy. *Real Algebraic Geometry*. Springer, 1998.

[29] D. Jibetean and E. de Klerk. Global optimization of rational functions: a semidefinite programming approach. *Mathematical Programming*, 106:93–109, 2006.

[30] N.K. Karmarkar and Y.N. Lakshman. On approximate GCDs of univariate polynomials. *Journal of Symbolic Computation*, 26(6):653–666, 1998.

[31] E. Kofidis and P. Regalia. On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, 23:863–884, 2002.

[32] Masakazu Kojima. Sums of squares relaxations of polynomial semidefinite programs. Technical Report B-397, Department of Mathematical and Computing Sciences Tokyo Institute of Technology, Tokyo, Japan, 2003.

[33] T. Kolda and B. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[34] T. Kolda and J. Mayo. Shifted power method for computing tensor eigenpairs. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1095–1124, 2011.

[35] S.G. Krantz and H.R.Parks. *The Implicit Function Theorem: History, Theory and Applications.* Birkhäuser,Boston, 2002.

[36] B. De Moor L. De Lathauwer and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[37] B. De Moor L. De Lathauwer and J. Vandewalle. On the best rank-1 and rank-$(R1, R2, ..., RN)$ approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[38] J.M. Landsberg. Tensors: geometry and applications. *Graduate Studies in Mathematics, American Mathematical Society, Providence, RI*, 128, 2012.

[39] Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[40] Jean B. Lasserre. *Moments, Positive Polynomials and Their Applications*. Imperial College Press, London, UK, 2009.

[41] Jean B. Lasserre. An algorithm for semi-infinite polynomial optimization. *TOP*, 20(1):119–129, 2012.

[42] Monique Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging Applications of Algebraic Geometry of IMA Volumes in Mathematics and its Applications*, volume 149, pages 157–270. Springer, 2009.

[43] C. Ling, J. Nie, L. Qi, and Y. Ye. Biquadratic optimization over unit spheres and semidefinite programming relaxations. *SIAM Journal On Optimization*, 20(3):1286–1310, 2009.

[44] Marco López and Georg Still. Semi-infinite programming. *European Journal of Operational Research*, 180(2):491–518, 2007.

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

[46] P.-A. Absil M. Ishteva and P. Van Dooren. Jacobi algorithm for the best low multilinear rank approximation of symmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, 34(2):651–672, 2013.

[47] Jiawang Nie. Sum of squares method for sensor network localization. *Computational Optimization and Applications*, 43(2):151–179, 2009.

[48] Jiawang Nie. Polynomial matrix inequality and semidefinite representation. *Mathematics of operations research*, 36(3):398–415, 2011.

[49] Jiawang Nie. Discriminants and nonnegative polynomials. *Journal of Symbolic Computation*, 47(2):167–191, 2012.

[50] Jiawang Nie. Sum of squares methods for minimizing polynomial forms over spheres and hypersurfaces. *Frontiers of Mathematics in China*, 7:321–346, 2012.

[51] Jiawang Nie. An exact Jacobian SDP relaxation for polynomial optimization. *Mathematical Programming, Ser. A*, 137:225–255, 2013.

[52] Jiawang Nie. Optimality conditions and finite convergence of Lasserre's hierarchy. *Mathematical Programming, Ser. A*, page To appear, 2014.

[53] Jiawang Nie. Certifying convergence of Lasserre's hierarchy via flat truncation. *Mathematical Programming, Ser. A*, to appear.

[54] Jiawang Nie, James Demmel, and Ming Gu. Global minimization of rational functions and the nearest GCDs. *Journal of Global Optimization*, 40(4):697–718, 2008.

[55] Jiawang Nie and Li Wang. Regularization methods for sdp relaxations in large scale polynomial optimization. *SIAM Journal On Optimization*, 22:408–428, 2012.

[56] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, 1999.

[57] L. Oeding and G. Ottaviani. Eigenvectors of tensors and algorithms for waring decomposition. *J. Symbolic Comput.*, 54:9–35, 2013.

[58] L.-H. Lim P. Comon, G. Golub and B. Mourrain. Symmetric tensors and symmetric tensor rank. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1254–1279, 2008.

[59] Panos M. Pardalos and Stephen A. Vavasis. Quadratic programming with one negative eigenvalue is NP-hard. *Journal of Global Optimization*, 1(1):15–22, 1991.

[60] Panos Parpas and Berç Rustem. An algorithm for the global optimization of a class of continuous minimax problems. *Journal of Optimization Theory and Applications*, 141(2):461–473, 2009.

[61] P. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2):293–320, 2003.

[62] P. Parrilo and B. Sturmfels. Minimizing polynomial functions. In *Proceedings of the DIMACS Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathe- matics and Computer Science (March 2001) (eds. S. Basu and L. Gonzalez-Vega)*, pages 83–100. American Mathematical Society, 2003.

[63] Mihai Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42:969–984, 1993.

[64] L. Qi. The best rank-one approximation ratio of a tensor space. *SIAM Journal on Matrix Analysis and Applications*, 32(2):430–442, 2011.

[65] B. Reznick. Some concrete aspects of Hilberts 17th problem. *Contemporary Mathematics*, 2000.

[66] R.T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Math. Oper. Res.*, 1(2):97–116, 1976.

[67] R.T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM J. Control and Optim.*, 14(5):877–898, 1976.

[68] M. Kojima S. Kim and H. Waki. Exploiting sparsity in SDP relaxation for sensor network localization. *SIAM Journal on Optimization*, 20(1):192–215, 2009.

[69] B. Savas and L.-H. Lim. Quasi-newton methods on grassmannians and multilinear approximations of tensors. *SIAM Journal on Scientific Computing*, 32(6):3352–3393, 2010.

[70] Konrad Schmüdgen. The K-moment problem for compact semi-algebraic sets. *Mathematische Annalen*, 289(1):203–206, 1991.

[71] Markus Schweighofer. Global optimization of polynomials using gradient tentacles and sums of squares. *SIAM Journal on Optimization*, 17(3):920–942, 2006.

[72] V. De Silva and L.-H. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1084–1127, 2008.

[73] Georg Still. Generalized semi-infinite programming: numerical aspects. *Optimization*, 49(3):223–242, 2001.

[74] Jos F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11/12:625–653, 1999.

[75] Yoshihiro Tanaka, Masao Fukushima, and Toshihide Ibaraki. A globally convergent SQP method for semi-infinite nonlinear optimization. *Journal of Computational and Applied Mathematics*, 23:141–153, 1988.

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

[77] K. C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3 - a MATLAB software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1998.

[78] Ha Huy Vui and Pham Tien Son. Global optimization of polynomials using the truncated tangency variety and sums of squares. *SIAM Journal on Optimization*, 19(2):941–951, 2008.

[79] Ha Huy Vui and Pham Tien Son. Solving polynomial optimization problems via the truncated tangency variety and sums of squares. *Journal of Pure and Applied Algebra*, 213(11):2167–2176, 2009.

[80] C. Ling X. Zhang and L. Qi. The best rank-1 approximation of a symmetric tensor and related spherical optimization problems. *SIAM Journal on Matrix Analysis and Applications*, 33(3):806–821, 2012.

[81] Xin yuan Zhao, Defeng Sun, and Kim chuan Toh. A newton-CG augmented lagrangian method for semidefinite programming. *SIAM Journal on Optimization*, 20(4):1737–1765, 2010.

[82] T. Zhang and G. H. Golub. Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, 23:534–550, 2001.

[83] Xinyuan Zhao, Defeng Sun, and Kim chuan Toh. SDPNAL version 0.1 – a MATLAB software for semidefinite programming based on a semi-smooth Newton-CG augmented Lagrangian method. http://www.math.nus.edu.sg/ mattohkc/SDPNAL.html.